PADERBORN UNIVERSITY
The University for the Information Society
Faculty for Computer Science, Electrical Engineering and Mathematics
Department of Computer Science
Research Group Codes and Cryptography

Master's Thesis

Submitted to the Codes and Cryptography Research Group
in Partial Fullfilment of the Requirements for the Degree of

Master of Science

Rational Secure Multiparty Computation

by
Henrik Bröcher

Thesis Supervisor:
Prof. Dr. Johannes Blömer

Paderborn, April 12, 2019

Erklärung

Ich versichere, dass ich die Arbeit ohne fremde Hilfe und ohne Benutzung anderer als der angegebenen Quellen angefertigt habe und dass die Arbeit in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegen hat und von dieser als Teil einer Prüfungsleistung angenommen worden ist. Alle Ausführungen, die wörtlich oder sinngemäß übernommen worden sind, sind als solche gekennzeichnet.

Ort, Datum	Unterschrift

Contents

1 Introduction

Since the seminal works of Yao [Yao86] and Goldreich, Micali, and Wigderson [GMW87] in the late 80s, the field of *secure Multiparty Computation (MPC)* has been one of the major topics in cryptography. It is about the *secure* evaluation of n–ary functions using protocols run by n parties with private inputs. Secure evaluation, informally, means that the parties learn nothing beyond their own outputs when engaging in the protocol even if a restricted number of parties is corrupted and colludes. Therefore, MPC protocols were often proven secure against adversaries which are allowed to corrupt and control some users in a certain (possibly arbitrary) manner while the other users remain honest. However, in 2004, Halpern and Teague [HT04] were the first to notice problems with existing MPC protocols, secure in this concept, when assuming rational protocol participants with selfish interests. Their considerations caught the interest of many researchers, leading to a lot of subsequent works and elegant ideas in the rather young field of *rational cryptography*.

In rational cryptography *every* party is assumed to act rational and selfish. This is modeled by a personal utility function for each party, for example, valuing outcomes of interactions with other parties. Rational parties choose strategies for the interaction in order to maximize their (expected) utility. Note, this is neither weaker nor stronger than the traditional model: On the one hand, no party acts arbitrarily like a corrupted one might do, but, on the other hand, no party is assumed to follow the protocol like an honest one. The task is to design protocols achieving desired (security) goals, while no party can do better by deviating from her prescribed strategy. Using game-theoretic notions, this relates to mechanism design where the prescribed strategies altogether form an equilibrium for the players.

As noted above, protocols which are provably secure in the traditional framework may fail in the rational model. To illustrate this, consider a (d, n)-secret-sharing scheme, i. e. a scheme where a dealer first distributes shares of a secret to n participants, such that the only way to recover (anything new about) the secret is knowing more than d valid shares. Letting each party broadcast her share is seemingly a simple protocol to make all parties recover the secret. This protocol suffices in the traditional framework as long as there are more than d honest parties and the malicious parties cannot send forged shares which alter the reconstructed secret. For a selfish party, however, there is no gain when broadcasting her share, since this at most enables different parties to learn the secret. Even worse, if exactly d other parties already did broadcast, the party can use her own share to recover the secret, but, by revealing her share, she makes the broadcasters learn the secret, too. This should lead to a decrease in utility since, usually, a secret is more valuable the less foreign parties know it. Summarizing, in one case not broadcasting is better than broadcasting and in the remaining cases not broadcasting is at

least not worse. Hence, a purely rational party rather chooses to send nothing instead of her share. In game-theoretic terms, the strategy of not broadcasting weakly dominates the strategy of broadcasting.

The problem above generalizes to any MPC protocol with a deterministically known last round r: In round r, selfish participants never gain but sometimes only lose from revealing *beneficial* information, i. e. information which at least in some cases helps *others* to learn a value of interest. Again, the strategy of sending such information is weakly dominated by not sending anything. Thus, rational parties will prefer the latter and not reveal any beneficial information in round r. Since this behavior is expected either from as well as by rational parties, round $r-1$ becomes the effective last round. For the same reasons as for round r no beneficial information is sent, either. By backwards induction this results in a scenario where the rational parties prefer to reveal no beneficial information in any round. In game theory, this process is called *iterated deletion of weakly dominated strategies (IDoWDS)*.

The inability of any deterministic-length MPC protocol to survive this process was noticed and proven by Halpern and Teague [HT04]. They propose to use protocols which overcome this problem by choosing the last round randomly during the execution. Particularly, they propose a secret reconstruction protocol for $n \geq 3$ players withstanding the IDoWDS. Gordon and Katz [GK06] elaborate on their results and propose a secret reconstruction protocol which, briefly, works as follows: In any round before the secret is revealed, the players *simultaneously* broadcast shares which either, with certain probability β, reconstruct the true secret or, with probability $1-\beta$, reconstruct an error value $\perp$. If any party P_i chooses not to broadcast her share in the current round, with probability β she obtains all shares necessary to reconstruct the secret, while the others lack her share. Assuming she has an incentive for exclusivity, she gains utility compared to the scenario where all parties learn the secret. However, with probability $1-\beta$ she reconstructs $\perp$ and revealed herself as deviating party. In this case, the parties are instructed to abort the protocol and, thus, a player P_i whose primary incentive is to learn the secret, loses compared to sticking to the protocol. If β is chosen such that the expected gain from exclusively learning the secret is outweighed by the expected loss from not learning it, this protocol can be shown to survive IDoWDS.

A common approach for realizing MPC protocols in the presence of rational parties is to, first, propose a protocol for secret reconstruction and prove its properties with respect to some game-theoretic model. Then, an intuition or description on how to use this protocol in the more general setting of rational MPC is given. For example, [HT04; GK06; Abr+06] follow this approach. *Intuitively*, this seems reasonable because secure MPC and the reconstruction of shared secrets are tightly coupled in the *traditional* model (c. f. [Gol04; CDN15]). However, they leave many details and subtleties on their concrete models and proofs implicit. This makes their theorems and corresponding proofs, proof sketches, or intuitions hard to analyze or extend in a *formal* way.

1.1 Related Work

In 2004, Halpern and Teague [HT04] proposed a formal way to analyze whether following some cryptographic MPC protocols is rational and realistic for participants. In particular, they demand the prescribed strategies of a protocol to form an equilibrium surviving the iterated deletion of weakly dominated strategies, which is still a common requirement. In this model they show the impossibility of fixed, finite length protocols for secret reconstruction and, more general, MPC protocols for parties with natural incentives for correctness and exclusivity. However, they overcome this result by proposing protocols for $n \geq 3$ players where the last round is chosen randomly. Gordon and Katz [GK06] elaborate on these results and provide a simpler protocol for

secret reconstruction which also work for two players. To achieve this, they remove an implicit assumption in the models and proofs of Halpern and Teague [HT04] which restricted the secret sharer in an unnecessary manner. Abraham et al. [Abr+06] then consider d-resilient equilibria, which, informally, requires that even coalitions of up to d parties gain nothing by deviating from prescribed strategies. They propose a protocol which employs, essentially, the same techniques as in [GK06] and prove its resiliency.

These works focus on rational secret-sharing and only sketch how to securely evaluate a function in the rational model. Halpern and Teague [HT04], for example, refer to the traditional, information-theoretically secure MPC protocol of Goldreich, Micali, and Wigderson [GMW87] and argue to replace the secret reconstruction phase by their own protocols. Although being reasonable, this and the other proposals lack full formal proofs. Furthermore, as noted recently by Garay et al. [Gar+13, Appendix B.2], these constructions rely on (implicit) assumptions regarding authenticity and privacy of communication channels. They point out that replacing ideal channels by computationally secure implementations is not as straightforward as in the traditional framework. Their concern is that a computationally hard problem can be broken after an (in the security parameter) exponential number of rounds. Thus, effectively, a fixed last round exists and, by the impossibility result of Halpern and Teague mentioned above, such a practical realization fails in theory.

Besides these results, rational secret sharing and MPC has been considered in different combinations of communication models and other equilibrial notions. For example, Fuchsbauer, Katz, and Naccache [FKN10] consider a model where no synchronous but only an asynchronous, point-to-point network is given. Halpern and Pass [HP10] consider a model where computational cost of a strategy is taken negatively into account when computing the final utility. This extends the notion of computational (Nash) equilibria as introduced by Dodis, Halevi, and Rabin [DHR00], where strategies form an equilibrium even if deviating negligibly increases the utility. A problem which all these protocols and approaches have in common is to require the exact values of the players' utilities regarding the possible outcomes or at least close approximations thereof. However, in reality these concrete values are not even known to the players themselves. Besides their useful model for rational secret-sharing, Asharov and Lindell [AL11] introduce a new notion called *utility independence* which overcomes this problem. They propose a protocol for secret reconstruction, which can be securely instantiated without relying on concrete utility values. Additionally, they prove several possibility and impossibility theorems regarding utility independence which is, essentially, only possible in settings where collusions of parties are restricted to be a strict minority.

Note, this thesis concentrates on analyzing cryptographic MPC protocols in a rational framework. In literature, the other way around, namely to instantiate game–theoretic models using cryptographic means, has been considered as well. Katz [Kat08] and Dodis and Rabin [DR07] provide broad overviews on important notions and refer to existing results within either direction.

1.2 Contributions

Within this thesis, we formally define a game-theoretic framework which is intended to facilitate a proper modeling of cryptographic problems. Although this framework and similar alternatives thereof have been sketched in several works before, its details were often left implicit, unclear, and, especially, not formalized. Based on this framework, we *formally* prove a more generalized version of a construction from Gordon and Katz [GK06] to build a function evaluation mechanism, i. e. a protocol which, informally, realizes a secure MPC protocol in a rational context. Additionally, we outline necessary and sufficient conditions under which this construction can be

instantiated in reality. Last but not least, motivated by an observation of Garay et al. [Gar+13, Appendix B.2], we *briefly* discuss the problems of IDoWDS within computational settings and describe an idea which *might* help to overcome these problems.

1.3 Structure

In Chapter 2, besides the notation used throughout this thesis, we define indistinguishability of random variables, introduce standard communication models used in cryptography, relevant basic notions from game theory, and the cryptographic concepts of secret-sharing and (traditionally) secure MPC. Next, in Chapter 3, we propose a concept reflecting computationally bounded players within a game-theoretic environment and model the cryptographic problems of secret reconstruction and function evaluation as games. Following, Chapter 4 contains our modularized construction for a protocol used for function evaluation which is proven to satisfy desired requirements under a set of certain assumptions. Based on that, Chapter 5 provides an overview on when and how these assumptions can or cannot be fulfilled. Eventually, in Chapter 6 we sum up our results and provide an overview of open problems which arose during writing this thesis and might be considered in subsequent works.

2 Preliminaries

In this chapter we provide an overview on the notation used throughout this thesis, followed by our definition of (non-uniform) indistinguishability of random variables. Next, we describe how interaction between machines is modeled, followed by a brief introduction of basic game-theoretical notions and their relations to the cryptographic context. Finally, we define secret-sharing schemes and the traditional framework of secure MPC.

2.1 Notation

- Function $\mu\colon \mathbb{N} \to \mathbb{R}$ is called *negligible* if for all $c \in \mathbb{N}$ there exists κ_c^*, such that for all $\kappa > \kappa_c^*$ we have $|\mu(\kappa)| < \kappa^{-c}$.
- For two functions $p_1, p_2\colon \mathbb{N} \to \mathbb{R}$ we write $p_1 \lesssim p_2$ if there exists a negligible function μ and $\kappa^* \in \mathbb{N}$ such that for all $\kappa > \kappa^*$ we have $p_1(\kappa) \leq p_2(\kappa) + \mu(\kappa)$. We write $p_1 \approx p_2$, if $p_1 - p_2$ is a negligible function.
- $[n] \coloneqq \{1, 2, \ldots, n\}$, where we implicitly assume $n \in \mathbb{N}$ here and in the following.
- For a set S the set of all subsets is $\mathcal{P}(S) \coloneqq \{X \mid X \subseteq S\}$.
- Let $I = \{i_1, \ldots, i_t\} \subseteq [n]$ be an index set in the following. We define $-I \coloneqq [n] \setminus I$.
- For a family of sets $\{S_i\}_{i\in[n]}$, let $\mathrm{S}_I \coloneqq S_{i_1} \times \cdots \times S_{i_t}$.
- Let $x_I \coloneqq (x_i)_{i\in I}$. Further $(x^1, \ldots, x^m)_I \coloneqq (x_i^1, \ldots, x_i^m)_{i\in I}$, e. g. in the following we use $(x, x', s)_I$ as more compact abbreviation for $(x_i, x_i', s_i)_{i\in I}$.
- For an *index* $i \in [n]$, let $x_{-i} \coloneqq x_{-\{i\}} = (x_i)_{i\in[n]\setminus\{i\}}$, i. e. the special case when excluding a single value from some tuple.
- For $x = (x_1, \ldots, x_n)$, let $(x_I, x_{-I}) \coloneqq x$. When the context is clear, we drop the additional parentheses for the sake of readability, e. g. $u(t_I, (x_I, x_{-I}))$ becomes $u(t_I, x_I, x_{-I})$.
- For a set S, let $\Delta(S)$ denote the set of all possible probability distributions on S.
- For a set S, let $s \leftarrow S$ denote the act of choosing an element *uniformly at random* from S and assigning it to s. For a distribution $\mathcal{D}$ (including output distributions of probabilistic Turing machines), let $s \leftarrow \mathcal{D}$ denote the act of sampling an element according to $\mathcal{D}$ and assigning it to s.

- For a random variable X, let $\mathrm{supp}(X) \coloneqq \{x | \Pr[X = x] > 0\}$ denote the support of X.
- For any distribution $\mathcal{D} \in \Delta(S_{[n]})$, any $(s_I, \cdot) \in \mathrm{supp}(\mathcal{D})$, let $D^{|s_I}$ denote the marginal distribution of $\mathcal{D}$ conditioned on s_I, i. e. for any $s_{-I} \in S_{-I}$ we have $\Pr[D^{|s_I} = (s_I, s_{-I})] \coloneqq \Pr[\mathcal{D} = (s_I, s_{-I}) | \mathcal{D} = (s_I, \cdot)]$.
- For any function $u \colon S \to \mathbb{R}$ and distribution $\mathcal{D} \in \Delta(S)$ let $u(\mathcal{D}) \coloneqq \mathbb{E}[u(\mathcal{D})]$, i. e. given a distribution $\mathcal{D}$ instead of concrete elements from its domain, any function in the following is mapped to its expected value with respect to $\mathcal{D}$.
- The set of all probabilistic, polynomial-time (ppt) interactive Turing machines is denoted by ITM.
- For ITMs $M = (M_1, \ldots, M_n)$, $\mathsf{Time}^M(x_{[n]}; r_{[n]})$ the runtime, in terms of of interaction rounds, on private input x_i and random-tape r_i for $i \in [n]$. Further, the expected runtime, $\mathsf{Time}^M(x_{[n]}) \coloneqq \mathbb{E}[\mathsf{Time}^M(x_{[n]}; r_{[n]})]$, where the expectation is taken over uniformly choosing the random-tapes r_i.

2.2 Indistinguishability of Random Variables

In cryptography we often use the concept of indistinguishability of random variables. Indistinguishability of two random variables Y_0, Y_1, intuitively, means that machines produce (almost) the same outputs when run on inputs distributed to either Y_0 or Y_1. In cryptography, the considered random variables often depend on some security parameter $\kappa \in \mathbb{N}$ or a string $\omega \in \Omega \subseteq \{0,1\}^*$. Particularly, for every κ or ω there exists a corresponding random variable $Y_b(\kappa)$ or, respectively, $Y_b(\omega)$ for $b \in \{0,1\}$. Hence, we rather consider *families of random variables* to be indistinguishable. Indistinguishability is generally defined with respect to a certain class of machines like, for example, probabilistic, polynomial-time (ppt) machines. Particularly, we call two families of random variables computationally indistinguishable if for any ppt machine the difference of its outputs is negligible in κ or the length $|\omega|$ of ω. If the difference of outputs is 0, then we call the families identically distributed. We define indistinguishability according to Goldreich [Gol01, Definition 3.2.7]. He uses the concept of families of polynomial-size circuits $\{C_n\}_{n\in\mathbb{N}}$ where every C_n corresponds to a Turing machine M_n whose size is polynomial in n and which runs in time polynomial in n for inputs of length n. This accounts for non-uniformity of distinguishers, i. e. distinguishers which for any input of size n may have a polynomial-size auxiliary input encoded. For further details on Boolean circuits and discussion of the notions we refer to the corresponding comprehensive descriptions within [Gol01].

Definition 2.1 (Indistinguishability)**.** *Let $\Omega \subseteq \{0,1\}^*$ and the families of random variables $\{Y_0(\omega)\}_{\omega\in\Omega}, \{Y_1(\omega)\}_{\omega\in\Omega}$. The families of random variables are* computationally indistinguishable*, denoted by $\{Y_0(\omega)\}_{\omega\in\Omega} \overset{\text{comp.}}{\equiv} \{Y_1(\omega)\}_{\omega\in\Omega}$, if for every family $\{C_n\}_{n\in\mathbb{N}}$ of polynomial-size circuits, every positive polynomial $p(\cdot)$, every sufficiently large n, and every $\omega \in \Omega \cap \{0,1\}^n$*

$$|\Pr[C_n(Y_0(\omega)) = 1] - \Pr[C_n(Y_1(\omega)) = 1]| < \frac{1}{p(n)}.$$

In case all differences equal 0, we say that the families are distributed identically, denoted by $\{Y_0(\omega)\}_{\omega\in\Omega} \overset{\text{id.}}{\equiv} \{Y_1(\omega)\}_{\omega\in\Omega}$.

Note, it suffices to consider random variables indexed by strings because for any natural number κ a random variable $Y(\kappa)$ can be identified uniquely by $Y(1^\kappa)$. Hence, the analogue definition for $\mathbb{N}$ as index set is implied. Sometimes we may only write $\{Y_b(\omega)\}_\omega$ when the set

of strings is clear by the context. Now, we can state a Lemma which becomes helpful during this thesis. Informally, if any function $u\colon \mathbb{N} \times Y \to \mathbb{R}$ which is polynomially bounded in its first input is fed with two computationally indistinguishable distributions over Y, the corresponding expected values differ at most negligibly. Recall, in our notation "$\approx$" means that two compared functions differ at most negligibly for sufficiently large κ. Formally, we defined $\approx$ only for functions with domain $\mathbb{N}$ but extend it to cover functions whose first inputs stem from $\mathbb{N}$.

Lemma 2.2. *Let $u\colon \mathbb{N} \times Y \to \mathbb{R}$ be a function and $\{Y_0(\kappa)\}_{\kappa\in\mathbb{N}}, \{Y_1(\kappa)\}_{\kappa\in\mathbb{N}}$ be two families of computationally indistinguishable random variables with* finite *domain Y. If u is polynomially bounded and positive, then*

$$\mathbb{E}[u(\kappa, Y_0(\kappa))] \approx \mathbb{E}[u(\kappa, Y_1(\kappa))].$$

Proof. We prove Lemma 2.2 by contradiction. In particular, we show how to construct a successful distinguisher for Y_0 and Y_1 if the expectation values' difference is non-negligible. Therefore, suppose there exists a polynomial $p \not\equiv 0$ such that for infinitely many $\kappa \in \mathbb{N}$ we have

$$|\mathbb{E}[u(\kappa, Y_0(\kappa))] - \mathbb{E}[u(\kappa, Y_1(\kappa))]| \geq p(\kappa). \tag{2.1}$$

If Equation (2.1) holds, then there exists at least one element $y \in Y$ in which the random variables differ polynomially. To formally prove this, we fist calculate

$$\begin{aligned}
|\mathbb{E}[u(\kappa, Y_0(\kappa))] - \mathbb{E}[u(\kappa, Y_1(\kappa))]| &= |\sum_{y\in Y} u(\kappa, y) \cdot \Pr[Y_0(\kappa) = y] - \sum_{y\in Y} u(\kappa, y) \cdot \Pr[Y_1(\kappa) = y]| \\
&= \sum_{y\in Y} u(\kappa, y) \cdot |\Pr[Y_0(\kappa) = y] - \Pr[Y_1(\kappa) = y]| \\
&\leq p'(\kappa) \sum_{y\in Y} |\Pr[Y_0(\kappa) = y] - \Pr[Y_1(\kappa) = y]|,
\end{aligned}$$

where p' is a polynomial which upper-bounds u's outputs and exists by assumption. Hence, using Equation (2.1), for infinitely many $\kappa \in \mathbb{N}$ we have

$$p'(\kappa) \sum_{y\in Y} |\Pr[Y_0(\kappa) = y] - \Pr[Y_1(\kappa) = y]| \geq p(\kappa) \Leftrightarrow \sum_{y\in Y} |\Pr[Y_0(\kappa) = y] - \Pr[Y_1(\kappa) = y]| \geq p(\kappa)/p'(\kappa).$$

Since p and p' are non-zero, positive polynomials, p/p' is either. Because Y is finite by assumption, there exists some y^* and polynomial p^*, such that for infinitely many κ we have

$$|\Pr[Y_0(\kappa) = y^*] - \Pr[Y_1(\kappa) = y^*]| \geq p^*(\kappa). \tag{2.2}$$

Otherwise, each term in the sum above would be bounded by a negligible function. However, summing up a finite number of negligible functions results, again, in a negligible function which grows slower than any polynomial and especially p/p'. Note, Y being finite is crucial as, otherwise, an infinite sequence of *non-repeating* values y^*_κ satisfying Inequality 2.2 could exist. Without loss of generality we assume $\Pr[Y_0(\kappa) = y^*] \geq \Pr[Y_1(\kappa) = y^*]$. We construct a distinguisher D which on input y^* outputs 0 and otherwise output $b \leftarrow \{0, 1\}$ uniformly at random. Corresponding to Definition 2.1, D can be seen as polynomial-size circuit making no use of its non-uniformity. This distinguisher D is ppt computable and has non-negligible advantage in

distinguishing the random variables, since for infinitely many $\kappa \in \mathbb{N}$ we have

$$\begin{aligned}
&\Pr[D(Y_0(\kappa)) = 0] - \Pr[D(Y_1(\kappa) = 0)] \\
&\quad = \sum_{i\in\{0,1\}} (-1)^i (\Pr[D(Y_i(\kappa)) = 0 \wedge Y_i(\kappa) = y^*] + \Pr[D(Y_i(\kappa)) = 0 \wedge Y_i(\kappa) \neq y^*]) \\
&\quad = \sum_{i\in\{0,1\}} (-1)^i (\Pr[D(Y_i(\kappa)) = 0 | Y_i(\kappa) = y^*] \Pr[Y_i(\kappa) = y^*] \\
&\qquad\qquad + \Pr[D(Y_i(\kappa)) = 0 | Y_i(\kappa) \neq y^*] \Pr[Y_i(\kappa) \neq y^*]) \\
&\quad = \sum_{i\in\{0,1\}} (-1)^i (1 \cdot \Pr[Y_i(\kappa) = y^*] + \frac{1}{2} \cdot (1 - \Pr[Y_i(\kappa) = y^*])) \\
&\quad = \sum_{i\in\{0,1\}} (-1)^i \frac{1}{2} \cdot \Pr[Y_i(\kappa) = y^*] \geq \frac{1}{2} p^*(\kappa),
\end{aligned}$$

where third equality holds, because, given y^*, D outputs 0 with probability 1 and in case $y \neq y^*$ it outputs 0 with probability $\frac{1}{2}$. However, since $\frac{1}{2}p^*$ is polynomial, this contradicts the computational indistinguishability of Y_0 and Y_1 and finishes the proof. □

2.3 Interactive Turing Machines and Communication Models

An interactive Turing machine (ITM) is an extension of Turing machines used to model interaction in protocol executions. Here, we characterize an ITM M by a randomized transition or next-messages function Next. Next takes common public input x, auxiliary private input a, state s and messages from other (uniquely identifiable) ITMs and outputs an updated state alongside messages for the other ITMs. The common input is intended for security parameters or public keys, whereas the auxiliary input may contain secret keys or, more general, models information gathered within the environment in which the machine was run. The states, for example, may contain all or a polynomial number of previously obtained messages.

The messages output by Next are delivered according to some specified communication model. These models have several dimensions like private/public, synchronous/asynchronous and simultaneous/non-simultaneous communication. For example, we sometimes assume a secure private channel between each pair of machines, an additional broadcast channel and synchronous, simultaneous communication. Synchronous and simultaneous means that all messages are delivered correctly from origin to receiver after *all* active machines have output their updated state and messages. Afterwards the next round begins and every machine is invoked with its updated state and received messages. The delivery can be modeled by a trusted party which, in every round $k \geq 1$, expects a (possibly empty) message $m_{i,j}^k$ from every active machine M_i to every active M_j and an additional broadcast message $m_{i,\mathrm{bc}}^k$. Only if the trusted party obtained all expected messages, it forwards them to the correct receivers. After an ITM finally halts it becomes inactive and announces this by a special halting symbol. Furthermore, every machine returns a final *local* output. The *transcript of an interaction* is the concatenation of *all* exchanged messages. The *view of machine* M_i, denoted by View_i, consists of its initial inputs followed by the (used) random bits and *all* its *received* messages. Note, that the uniformly chosen random bits and (possibly randomly chosen) initial inputs induce a probability distribution on the occurring transcripts, views and final (local) outputs. In particular, for given machines $M = (M_1, \ldots, M_n)$ and respective private inputs $t_1, \ldots, t_n$, we refer to these distributions by the random variables $\mathcal{T}^M(t_1, \ldots, t_n)$, $\mathsf{View}^M(t_1, \ldots, t_n)$ and $\mathsf{Out}^M(t_1, \ldots, t_n)$.

Sometimes we need to restrict ITMs with respect to their running time. An ITM M is ppt computable, if there exists a polynomial $t\colon \mathbb{N} \to \mathbb{N}$, s. t. on common input x, Next is computable within $t(|x|)$ steps. Note, that the condition depends only on the common input which guarantees that interacting ITMs have polynomially related running times, independent of their other inputs. Although messages and states are computed in polynomial time, the number of rounds until interaction ends can be exponential or even unbounded in the (common) input size. Indeed, in the context of *rational* cryptography protocols, generally, must not have a fixed upper bound on the number of rounds. To remain feasible in practice, we may require the *expected* number of rounds as constant or polynomially bounded in the input size. Note, that a ppt ITM can only output states of polynomial size (in its provided input length). Hence, in protocols with no round-limit it is always possible to reach a round, where not all previously obtained messages can be stored in a state. This can be overcome by allowing Next in round k to run in time polynomial in $|x| + k$, called liberal ppt by Katz [Kat08].

2.4 Game Theory

The broad field of game theory concentrates on modeling and explaining the behavior of rational players within strategic games. In the following we introduce basic notions used within this thesis and describe their relations to the cryptographic context. We stick close to the definitions of Katz [Kat08] due to their good extensibility to game theory within a computational context (Section 3.1). For full introductions into game theory we refer to Osborne and Rubinstein [OR94] or Blumrosen and Nisan [BN07]. We begin with the definition of a normal form game.

Definition 2.3 (Normal Form Game)**.** *A* normal form game $\Gamma = \left(\{S_i\}_{i\in[n]}, \{u_i\}_{i\in[n]}\right)$ with n players $P_1 \ldots, P_n$ *consists of a* set of strategies S_i *and a* utility function $u_i\colon S_{[n]} \to \mathbb{R}$ *for every player* P_i*. We call* $s_{[n]} \in S_{[n]}$ pure strategy profile *and* $u_i(s_{[n]})$ *the* utility of player P_i *given* $s_{[n]}$*.*

In a normal form game, every player P_i chooses a strategy s_i from her personal set of strategies S_i. The resulting pure strategy profile $s_{[n]}$ then yields utility $u_i(s_{[n]})$ to player P_i. P_i's choice of strategy s_i might be randomized, but is assumed to be independent of the remaining players' choices. To model this choice process we introduce mixed strategies.

Definition 2.4 (Mixed Strategies)**.** *For normal form game* $\Gamma = \left(\{S_i\}_{i\in[n]}, \{u_i\}_{i\in[n]}\right)$ *we call* $(\sigma_1, \ldots, \sigma_n)$ *with* $\sigma_i \in \Delta(S_i)$ *a* (mixed) strategy profile*.* P_i*'s utility for a (mixed) strategy profile is* $u_i(\sigma_{[n]}) \coloneqq \mathbb{E}[s_{[n]} \leftarrow \sigma_{[n]} : u_i(s_{[n]})]$*.*

By Definition 2.4, a mixed strategy σ_i of P_i is a distribution over P_i's personal strategies. In this way, mixed strategies incorporate the meta-level of choosing a strategy into the game theoretic model. For a given profile $\sigma_{[n]}$, P_i's utility $u_i(\sigma_{[n]})$ is her average utility, i. e. the expected utility over every player independently sampling a strategy $s_i \leftarrow \sigma_i$ and then obtaining $u_i(s_{[n]})$. Using P_i's expected utility for a given profile accounts for *risk-neutral* players who, for example, do not care about the corresponding variance. A risk-neutral player, for example, is indifferent between taking a strategy which yields 50$ always and a strategy which yields 0$ or 100$ with probability $\frac{1}{2}$ each.

Another way of expressing games are extensive form games which, essentially, are designed to explicitly reflect games consisting of several rounds where players choose their strategies sequentially. However, every game in extensive form can be reduced to one in normal form, for example, by using ITMs. As an example consider the game of chess which is played sequentially. *Pure* strategies can be modeled as *deterministic* ITMs (M_1, M_2) where P_1's strategy M_1 initiates the interaction by sending a valid move to M_2. The machines then, alternating, send their moves

based on the previous moves until the game ends. P_i's utility $u_i(M_1, M_2)$ is then defined on whether the machines (deterministic) interaction transcript represents a win, tie or loss for P_i. In this modeling mixed strategies correspond to probabilistic ITMs M_i from which a pure strategy is sampled by choosing M_i's random tape uniformly at random. The utility for probabilistic ITMs, thus, would be the expected utility over the distribution of transcripts.

In the above sense, normal form games are suited to model rational MPC where the players are allowed to interact in order to evaluate a given function. However, until now we have no means to provide every party her private input x_i for which they want to compute $f(x_{[n]})$. Indeed, the whole game description and its state are common knowledge among the players. Games where this is the case are said to be of *complete information*. Private inputs are formally modeled by *typed games* where at the game's beginning *types* are sampled and privately given to the players. Note, in the following we make use of our notational convention that a function which is given distributions over its domain rather elements thereof maps to its expected value according to these distributions.

Definition 2.5 (Typed Game)**.** *A* typed game $\Gamma = \left(\{S_i\}_{i\in[n]}, \{u_i\}_{i\in[n]}, \mathcal{D}\right)$ *consists of a* type distribution $\mathcal{D} \in \Delta(T_{[n]})$ *over finite sets of types* T_i*, strategy sets* S_i *and utility functions* $u_i \colon T_{[n]} \times S_{[n]} \to \mathbb{R}$*. At the beginning of a typed game the types* $t_{[n]} \leftarrow \mathcal{D}$ *are sampled and every type* t_i *is privately given to the corresponding player* P_i*. Before obtaining her type,* P_i*'s (a-priori) utility for strategy profile* $\sigma_{[n]}$ *is* $u_i(\sigma_{[n]}) \coloneqq u_i(\mathcal{D}, \sigma_{[n]})$*, i. e.* P_i*'s expected utility over the distributions of* $\mathcal{D}$ *and* $\sigma_{[n]}$*. After obtaining type* t_i*,* P_i*'s (a-posteriori) utility is* $u_i(t_i, \sigma_{[n]}) \coloneqq u_i(\mathcal{D}^{|t_i}, \sigma_{[n]})$ *where* $\mathcal{D}^{|t_i}$ *is* $\mathcal{D}$ *conditioned on type* i *being* t_i*.*

Remark. *Every normal form game* $\Gamma = \left(\{S_i\}_{i\in[n]}, \{u_i\}_{i\in[n]}\right)$ *can be described as typed game using a type distribution* $\mathcal{D}$ *with* $\Pr[\mathcal{D} = t_{[n]}] = 1$ *for some "dummy" types* $t_{[n]}$ *and utilities defined by* $u_i(t_{[n]}, \sigma_{[n]}) \coloneqq u_i(\sigma_{[n]})$*. In this sense, the following definitions apply to normal form games analogously.*

As a first example for a typed game think of a card game. In the beginning a dealer distributes the cards uniformly at random among the players. The cards correspond to types and every P_i can choose strategies depending on the game's rules *and* the obtained cards. Note, type distribution $\mathcal{D}$ is from $\Delta(T_{[n]})$ and not $\bigtimes_{i\in[n]} \Delta(T_i)$ which means that types are not necessarily independent but possibly correlated. In the card game, for example, every player can rule out the possibility that another participant obtained the cards she already owns. Because players might deduce information *conditioned* on their given type such games are also called *Bayesian Games* [OR94]. In the context of rational MPC types are used to model the players' private inputs x_i on which they want to jointly evaluate $f(x_{[n]})$. These values might be correlated, for instance, in some situations a player P_i possibly knows that another player P_j's value x_j is at most twice as big as her value x_i.

Until now the main focus of our definitions lies upon the modeling of games. Another important aspect of game theory is the search and explanation of stable situations, i. e. situations where no player has an incentive to change her strategy. Strategy profiles where such a property holds are called equilibria.

Definition 2.6 (Typed Equilibrium)**.** *For typed game* $\Gamma = \left(\{S_i\}_{i\in[n]}, \{u_i\}_{i\in[n]}, \mathcal{D}\right)$ *we call strategy profile* $(\sigma_1, \ldots, \sigma_n)$ typed equilibrium *if for all* $i \in [n]$ *and pure strategies* $s_i' \in S_i$

$$u_i(\mathcal{D}, \sigma_{[n]}) \geq u_i(\mathcal{D}, s_i', \sigma_{-i}).$$

Definition 2.6 ensures that no player P_i can exchange her strategy and thereby increase her utility with respect to type distribution $\mathcal{D}$. A stronger version, for example, requires that for

any t_i with $\Pr[\mathcal{D} = (t_i, \cdot)] > 0$ we have $u_i(t_i, \sigma_{[n]}) \geq u_i(t_i, s'_i, \sigma_{-i})$. However, both notions are equivalent for the games within this thesis. Particularly, we consider strategies to correspond to ITMs. Every ITM M_i in use can have a different strategy $M_i^{t_i}$ for every type t_i from the *finite* set T_i encoded. If we have $u_i(t_i^*, M_i^{t_i}, M_{-i}) > u_i(t_i^*, M_{[n]})$ for some $t_i^* \in T_i$ then ITM M'_i which runs $M_i^{t_i^*}$ for t_i^* and M_i otherwise yields

$$\begin{aligned} u_i(M'_i, M_{-i}) &= \sum_{t_i \in T_i} u_i(t_i, M'_i, M_{-i}) \\ &= u_i(t_i^*, M_i^{t_i^*}, M_{-i}) + \sum_{t_i \in T_i \setminus \{t_i^*\}} u_i(t_i, M_i, M_{-i}) \\ &> u_i(t_i^*, M_i, M_{-i}) + \sum_{t_i \in T_i \setminus \{t_i^*\}} u_i(t_i, M_i, M_{-i}) \\ &= u_i(M_{[n]}). \end{aligned}$$

Hence, if $M_{[n]}$ is an equilibrium according to Definition 2.6, t_i^* must not exist. In the rational MPC context such a typed equilibrium, especially, ensures that a *single rational* player cannot gain utility by deviating from her strategy. The difference to traditional MPC is, that players are not enforced to stick to a given strategy solely by cryptographic means but also by their utilities. We generalize Definition 2.6 to ensure that a limited number of colluding players cannot replace their strategies to gain utility, either.

Definition 2.7 (d-Resilient Equilibrium)**.** *For typed game* $\Gamma = \left(\{S_i\}_{i\in[n]}, \{u_i\}_{i\in[n]}, \mathcal{D}\right)$ *and* $1 \leq d < n$ *we call strategy profile* $(\sigma_1, \ldots, \sigma_n)$ d-resilient equilibrium *if for all coalitions of players* $C \subseteq [n]$ *with* $|C| \leq d$*, all* $s'_C \in S_C$ *and all* $i \in [n]$ *it holds that*

$$u_i(\mathcal{D}, \sigma_{[n]}) \geq u_i(\mathcal{D}, s'_C, \sigma_{-C}).$$

Definition 2.7 ensures that it is impossible for up to d players to collectively choose a strategy s'_C and increase at least one of the colluding players' utilities. This is a strong notion of d-resiliency, because one gaining player suffices and the remaining player's utilities might decrease arbitrarily. Another reasonable definition could, for example, focus on increasing the coalition's average utility. However, Definition 2.7 better suits the cryptographic idea of having a single (adversarial) players who controls the colluding players.

Equilibria are stable in the sense that players or coalitions cannot *increase* their utility by another strategy assuming the remaining players stick to their strategy. Nevertheless, there might exist strategies which always yield the same utility as the current strategy and in some situations even higher utility. Such strategies are called weakly dominating and are defined as follows.

Definition 2.8 (Weak Domination)**.** *Strategy* $s_i \in S_i$ *is* weakly dominated with respect to S_{-i} *if there exists* $s'_i \in S_i$*, such that (1) for all* $s_{-i} \in \mathrm{S}_{-i}$ *we have* $u_i(\mathcal{D}, s'_i, s_{-i}) \geq u_i(\mathcal{D}, s_i, s_{-i})$ *and (2) there exists* $s'_{-i} \in \mathrm{S}_{-i}$ *such that* $u_i(\mathcal{D}, s'_i, s_{-i}) > u_i(\mathcal{D}, s_i, s_{-i})$*. The* set of weakly dominated strategies in S_i with respect to S_{-i} *is denoted by* $\mathrm{DOM}(\Gamma, i)$*.*

It is reasonable that rational players use weakly domination as tie breaker for two strategies which yield the same utility within a given situation. Indeed, in reality a strategy might never be chosen and, thus, can be ignored and deleted from the game if it is weakly dominated by another strategy. However, when for some strategy s_i and a corresponding s'_i all strategies contradicting condition (1) are deleted, s_i becomes weakly dominated by s'_i and, thus, can be excluded. This consideration results in the following process of iterated deletion of weakly dominated strategies.

Definition 2.9 (Iterated Deletion of Weakly Dominated Strategies (IDoWDS))**.** *Let typed game* $\Gamma^0 = \left(\{S_i^0\}_{i\in[n]}, \{u_i\}_{i\in[n]}, \mathcal{D}\right)$*. For* $k \geq 1$*, define* $S_i^k := S_i^{k-1} \setminus \mathrm{DOM}(\Gamma^{k-1}, i)$ *and* $\Gamma^k = \left(\left\{S_i^k\right\}_{i\in[n]}, \{u_i\}_{i\in[n]}, \mathcal{D}\right)$*. Let* $S_i^\infty := \cap_{k=0}^\infty S_i^k$*. A typed equilibrium* $(\sigma_1, \ldots, \sigma_n)$ survives IDoWDS*, if* $\sigma_i \in \Delta(S_i^\infty)$ *for all* $i \in [n]$*.*

This process, especially, was applied by Halpern and Teague [HT04] as follows to reason that traditional MPC (and secret reconstruction) protocols generally do not work assuming rational players: In traditional (unfair) MPC protocols there always exists a last round r where new information is sent by some players' strategies, i. e. ITMs. Since r is the last round, not sending the message makes no difference for any sending player P_i's outputs. Moreover, if we assume that P_i's utility does only decrease when other players learn more about the outputs, it might even be worse to reveal new information in round r Hence, ITMs which send a "useful" message in round r are weakly dominated by ones not sending anything and are deleted according the above process. However, in the resulting game there also exists a last round $r' < r$ with useful information. By backwards induction the same reasoning is applied and the deletion continues until no strategies revealing "useful" information remain.

As we have seen, in an equilibrium no players or coalitions have incentives do deviate from their strategy *assuming* the remaining players stick to their given strategies. A natural question is, how the players might come to such an assumption or state. In a *mechanism* this knowledge is established by announcing an equilibrium alongside the game.

Definition 2.10 (Practical Mechanism)**.** *For a typed game* Γ *and a strategy profile* $\sigma_{[n]}$ *we call* $(\Gamma, \sigma_{[n]})$ *a* d-resilient mechanism*, if* $\sigma_{[n]}$ *is a* d*-resilient equilibrium for* Γ*. We, additionally, call it* practical *if* $\sigma_{[n]}$ *survives IDoWDS in* Γ*.*

In other words, if we have a d-resilient practical mechanism we may assume the *rational* size-d coalitions to stick to their prescribed strategies. The field of designing such mechanisms is similar to protocol design in cryptography. Indeed, mechanisms can be considered as protocol as follows: Recommended strategies correspond to the protocol description and alternative strategies display allowed deviations of protocol participants. In this sense, (practical) mechanisms correspond to protocols from which rational participants do not deviate based on their utilities. In traditional cryptography, protocols are designed such that a restricted number of arbitrarily acting parties cannot break chosen security properties as long as the others stick to the protocol.

In game theory it sometimes is meaningful to augment games by a trusted third party called mediator. Such a mediator is similar to an ideal world functionality in traditional cryptographic MPC models (c. f. Section 2.6). Mediator M privately recommends centrally sampled and possibly correlated strategies to the players. In typed games, M's choices might depend on the player's types or, more precise, on the types the players disclose to M. In rational cryptography, we will use mediated games to characterize settings where, for example, no strategy profile which leads all parties to the desired outcome may exists.

Definition 2.11 (Mediated Typed Game)**.** *A* mediated typed game $\Gamma = \left(\{S_i\}_{i\in[n]}, \{u_i\}_{i\in[n]}, \mathcal{D}, \mathcal{M}\right)$ *is a typed game (Definition 2.5) with type sets* $T_1, \ldots, T_n$ *and an additional* mediator strategy $\mathcal{M}\colon T_{[n]} \to \Delta(S_{[n]})$*. After every player* P_i *privately obtains her type* t_i*, she privately sends some* $t_i' \in T_i$ *to mediator* M*. Mediator* M *then samples* $s_{[n]} \leftarrow \mathcal{M}(t_{[n]}')$ *and privately returns* (t_i', s_i) *to the corresponding player* P_i*. Finally, every player* P_i *chooses and plays a strategy* $s_i' \in S_i$*.*

In a mediated typed game P_i*'s (effective) strategies are* $\delta_i = (\delta_i^1, \delta_i^2)$ *with* $\delta_i^1\colon T_i \to \Delta(T_i)$ *and* $\delta_i^2\colon T_i \times T_i \times S_i \to \Delta(S_i)$*. For a given strategy profile* $(\delta_1, \ldots, \delta_n)$*,* P_i*'s utility is the expected utility over all random choices within the process described above. Formally,*

$$u_i(\mathcal{D}, \delta_{[n]}) := \mathbb{E}[t_{[n]} \leftarrow \mathcal{D}, t_{[n]}' \leftarrow (\delta_j^1(t_j))_{j\in[n]}, s_{[n]} \leftarrow \mathcal{M}(t_{[n]}'), s_{[n]}' \leftarrow (\delta_j^2(t_j, t_j', s_j))_{j\in[n]} : u_i(t_{[n]}, s_{[n]}')].$$

Hence, in a mediated version of a typed game, the players have two choices, which are modeled by their strategies (δ^1, δ^2). Their first choice is about the concrete (possibly substituted) type sent to the mediator, while her second choice is whether to follow the recommendation or choose a different strategy. In some scenarios a player may has an incentive to lie about her type in order to make the other parties who input their true types learn worse recommendations. This is often the case when, based on the returned recommendation, P_i can compute the recommendation she would have obtained when inserting her true type. Note, the mediator returns every party her inserted type in order to prevent the modeling of internal states on the players' sides. The strategy when a player always inserts her true type and always follows M's recommendation is called canonical strategy.

Definition 2.12 (Canonical Strategy)**.** *Let mediated typed game* $\Gamma = \left(\{S_i\}_{i\in[n]}, \{u_i\}_{i\in[n]}, \mathcal{D}, \mathcal{M}\right)$ *(Definition 2.11). We call strategy* $\delta_i^* = (\delta_i^1, \delta_i^2)$ *P_i's* canonical strategy*, if for all* $t_i \in T_i$ *and* $s_i \in S_i$ *we have* $\Pr[\delta_i^1(t_i) = t_i] = \Pr[\delta_i^2(\cdot, \cdot, s_i) = s_i] = 1$.

If playing the canonical strategy is a (d-resilient) eqilibrium, then a given setting often has a desirable property. In Sections 3.2 and 3.3 we will see concrete examples how these mediated games and equilibria can be used.

2.5 Secret-Sharing

Secret-sharing schemes were independently introduced by Shamir [Sha79] and Blakley [Bla79]. They, in general, are used to share some secret information among parties $P_1, \ldots, P_n$ such that only explicitly defined groups thereof can reconstruct the secret. Furthermore, they are an important tool for constructing traditionally secure MPC protocols as defined in Section 2.6. Beimel [Bei11] provides a detailed survey on existing schemes and notions which goes beyond the scope of this thesis.

In secret-sharing schemes the secret is shared among a set of parties $P_1, \ldots, P_n$. The groups of parties qualified to reconstruct secrets are defined by so-called access structures.

Definition 2.13 (Access Structure)**.** *Let* $M = \{P_1, \ldots, P_n\}$ *be a set of n parties. A collection* $\mathcal{A} \subseteq \mathcal{P}(M)$ *is called* monotone *if* $A \in \mathcal{A}$ *and* $A \subseteq B \subseteq M$ *implies* $B \in \mathcal{A}$*. An* access structure $\mathcal{A} \subseteq \mathcal{P}(M)$ with n parties *is a* monotone collection *of* non-empty *subsets of* M*. A* (d, n)*-access structure with threshold* $0 \leq d < n$ *is defined as* $\mathcal{A} \coloneqq \{A \subseteq M \mid d < |A|\}$*. A set* $A \subseteq M$ *is called* qualified *if* $A \in \mathcal{A}$ *and* non-qualified *if* $A \notin \mathcal{A}$*.*

By Definition 2.13, any access structure is monotone, meaning, if any party joins a qualified group the resulting group remains qualified. An access structure with threshold d ensures that all groups of at least $d+1$ parties are qualified to reconstruct secrets. This property is very useful when talking about security against attackers allowed to control up to d parties. A secret-sharing scheme is always related to an access-structure and defined as follows:

Definition 2.14 (Secret-Sharing Scheme)**.** *Let* $\mathcal{A}$ *be an access structure with n parties and S be a finite set of secrets where* $|S| \geq 2$*. A* (perfect) secret-sharing scheme with domain of secrets S realizing access structure $\mathcal{A}$ *is a tuple of ppt algorithms* $(\mathsf{Share}, \mathsf{Recon})$*, where*

1. *On input secret* $s \in S$*,* Share *outputs a tuple of shares* $s_{[n]} \in S_{[n]}$ *where* S_i *is called* domain of shares of party P_i*.*

2. *Correctness:* Recon deterministically *outputs an element from* $S \cup \{\perp\}$ *such that for all* $s \in S$ *and* $A \in \mathcal{A}$

$$\Pr[\mathsf{Recon}(\mathsf{Share}(s)_A) = s] = 1.$$

3. *Perfect Privacy: For all non-qualified sets $A \notin \mathcal{A}$ and every two secrets $s, s' \in S$, we have* $\mathsf{Share}(s)_A \overset{\text{id.}}{\equiv} \mathsf{Share}(s')_A$.

Definition 2.15 ((d,n)-Secret-Sharing Scheme)**.** *A (d,n)-secret-sharing scheme is a (perfect) secret-sharing scheme (Definition 2.14) with a (d,n)-access structure.*

The idea of a secret-sharing scheme is that a secret s is determined, then a sharing $s_{[n]} \leftarrow \mathsf{Share}(s)$ is sampled and, eventually, every P_i is privately given her share s_i. Privacy ensures that any *non-qualified* group learns, information-theoretically, nothing about the shared secret s by pooling their shares. In other words, the shares obtained by any non-qualified group are independent of the shared secret itself. On the other hand, correctness guarantees that any qualified group can always reconstruct the secret s by pooling their shares and using Recon.

Note, correctness only applies to situations where every party P_i provides her true share s_i. However, for example, suppose an $(n-1,n)$-secret sharing scheme, i. e. all n parties need to pool their shares in order to reconstruct secret s. Further, assume parties $P_1, \ldots, P_{n-1}$ already published their shares. By perfect privacy and correctness for any secret s' there exists a share s'_n such that $\mathsf{Recon}(s_{-n}, s'_n) = s'$, i. e. P_n might replace her share and choose the reconstructed secret s' while herself can (exclusively) compute $\mathsf{Recon}(s_{-n}, s_n) = s$. Hence, in settings with rational players having an incentive for exclusivity, reconstruction is impossible as long as replacing shares is possible. To prohibit the unnoticed replacement of up to d shares in the presence of (at least) one honest party, we introduce the notion of d-authenticated secret-sharing schemes. Because we distinguish perfect and computational d-authentication, we slightly change the syntax of Share in the following by adding 1^κ. This allows ppt Share to run in time polynomial in κ. Further we assume that every share s_i is of size at least κ, allowing adversary A and Recon runtime which is polynomial in κ, too.

Definition 2.16 (d-Authenticated Secret-Sharing)**.** *A secret-sharing scheme* $(\mathsf{Share}, \mathsf{Recon})$ *is called* computationally d-authenticated *if for all secrets $s \in S$, all coalitions $C \subseteq [n]$ of size $|C| \leq d$, all $I \subseteq [n] \setminus C$ with $I \neq \emptyset$ and all ppt A we have*

$$\Pr[s_{[n]} \leftarrow \mathsf{Share}(1^\kappa, s), (s'_C) \leftarrow A(s_C) : s'_C \neq s_C \wedge \mathsf{Recon}(s'_C, s_I) \neq \perp] \approx 0.$$

If the probability equals 0 *for all* (unbounded) *adversaries A we say* $(\mathsf{Share}, \mathsf{Recon})$ *is* perfectly d-authenticated.

In other words, if a secret-sharing scheme is computationally d-authenticated and at least one party honestly inserts her share, no ppt adversary can feasibly insert new share(s) without Recon indicating this by $\perp$. As sketched above, in the presence of rational parties this property is necessary with respect to reconstructing a secret because, otherwise, parties or coalitions gaining from exclusivity would always exchange their shares. If the parties are not bounded computationally, we need to require perfect authentication. Note, this notion is different from verifiable secret-sharing ([Cho+85]) where the sharing instance might be assumed as malicious, either. Yet, a similar notion is that of δ-robust secret-sharing (e. g. [Che15]).

However, for any $d < n$ any (d,n)-secret-sharing scheme can be augmented to fulfill Definition 2.16 assuming, for example, secure digital signature schemes. Digital signature schemes and their security date back to the ideas of Diffie and Hellman [DH76]. They ensure the infeasibility of forging valid signatures without access to the sampled secret key while a corresponding public key enables to check a signature's validity. The augmented secret-sharing scheme's method $\mathsf{Share}'(s)$, besides sampling shares from $\mathsf{Share}(s)$, generates signing and verification key and signs each share. Every P_i obtains an augmented share consisting of the share itself, its signature and the corresponding public key. The new reconstruction method Recon' runs the old Recon

only if all included public keys are the same and every signature is valid; Otherwise it outputs $\perp$. Note, unconditional perfect d-authentication with efficient reconstruction is only possible if $d < \frac{n}{3}$ and impossible for $d \geq \frac{n}{2}$ [Che15]. If we drop the assumption for efficient reconstruction, then $d < \frac{n}{2}$ is possible, unconditionally, by checking whether all qualified subsets of sent shares reconstruct to the same secret. Note, if n shares are sent, this exhaustive search requires a number of checks which is exponential in n. To the best of our knowledge, for $\frac{n}{2} > d > \frac{n}{3}$ and efficient reconstruction there is always a trade-off with respect to accuracy of detecting forged shares and remaining privacy.

2.6 Secure Multipary Computation

In Multiparty Computation (MPC) we establish models and protocols suited for settings where several parties possess private inputs on which they jointly evaluate functions. Security, generally, corresponds to computing correct results while preserving the inputs' privacy up to information revealed by the results themselves. Within this section we establish the traditional MPC security model from Goldreich [Gol04, Section 7]. Later in this thesis we employ some secure protocol as subcomponent to construct a protocol for securely evaluating functions in a game-theoretic MPC model. For more detailed explanations and discussions of the following notions we refer to Goldreich [Gol04, Section 7].

Before considering security itself, we explain which kind of functions we want to evaluate. Indeed, we allow *functionalities* f which can be seen as randomized functions. Particularly, an *n-ary functionality* maps n inputs $x_{[n]}$ to the *random variable* $f(x_{[n]})$ and not a simple value. A requirement for an MPC protocol is to have the identical output distribution as f when fed with the same inputs and run honestly.

Definition 2.17. *Let $f\colon (\{0,1\}^*)^n \to (\{0,1\}^*)^n$ be an n-ary functionality. A protocol $\Pi = (\Pi_1, \ldots, \Pi_n)$* computes *$f$ if for all $x_{[n]} \in (\{0,1\}^*)^n$ we have $\mathsf{Out}^{\Pi}(x_{[n]}) \overset{\mathsf{id.}}{\equiv} f(x_{[n]})$ and Π runs in expected polynomial time.*

Hence, a protocol which computes f guarantees Π's outputs to be distributed correctly *if all parties honestly run* $\Pi_i(x_i)$. Yet, this ensures no further security properties. For instance, a trivial protocol satisfying Definition 2.17 instructs every Π_i on input x_i to send x_i to Π_1 which, then, samples $y_{[n]} \leftarrow f(x_{[n]})$ and returns every y_i to the corresponding Π_i. Obviously, the party running Π_1 learns all the inputs and, additionally, is able to undetectably broadcast any incorrect value y_i' which could result from the input x_i. However, this contradicts both, the privacy of inputs as well as the correctness of outputs.

We continue by formalizing when we consider protocol Π computing functionality f *(actively) secure* in the traditional cryptographic model. Following approach relies on the ideal/real world paradigm. Using this paradigm we, essentially, show that *any* possible adversarial behavior against Π (in the real world) corresponds to inevitable and allowed adversarial behavior within the defined ideal world. This ideal world is based on non-realizable assumptions. In particular, we assume the existence of an *incorruptible* trusted third party for sampling f's outputs. Additionally, every party P_i possesses a *private* communication channel to this third party. An adversary B in our ideal world is assumed to control a *static* set $C \subseteq [n]$ of parties. The parties not in C, called honest, send their true inputs to the third party and output whatever it returns. The adversary, however, is allowed to deviate in the following ways: B may *substitute the inputs x_C of C by any x_C'* before sending them to the trusted party. Moreover, if controlling some special party (here 1), after obtaining her values sampled according to $f(x_C', x_{-C})_C$ B may *prematurely abort* which makes the non-corrupted parties output $\perp$. B's final (local) output itself

may be any function on all information and inputs it obtained. In all these steps B may use some given auxiliary input modeling, for example, knowledge from previous protocol executions.

Definition 2.18 (Ideal MPC Model)**.** *Let $f\colon (\{0,1\}^*)^n \to (\{0,1\}^*)^n$ be an n-ary functionality and pair (C,B), where $C \subseteq [n]$ and B is a ppt algorithm representing an adversary in the ideal model. The joint execution of f under (C,B) in the ideal model (on input $x = (x_1,\dots,x_n)$ and auxiliary input z), denoted $\mathsf{Ideal}_{f,C,B(z)}(x)$ is defined by uniformly selecting a random-tape r for the adversary and letting $\mathsf{Ideal}_{f,C,B(z)}(x) \coloneqq \Upsilon(x,C,z,r)$, where $\Upsilon(x,C,z,r)$ is defined for $x'_C \coloneqq B(x_C,C,z,r)$ and $x' \coloneqq (x'_C, x_{-C})$ as follows:*

- *If $1 \notin C$, $\Upsilon(x,C,z,r) \coloneqq (f(x')_{-C}, B(x_C,C,z,r,f(x')_C)$.*
- *If $1 \in C$, $\Upsilon(x,C,z,r) \coloneqq \begin{cases} (\bot^{|-C|}, B(x_C,C,z,r,\mathrm{f}(x')_C,\bot)), & \text{if } B(x_C,C,z,r,f(x')_C) = \bot \\ (f(x')_{-C}, B(x_C,C,z,r,f(x')_C), & \text{otherwise.} \end{cases}$*

Definition 2.18 reflects the capabilities of an ideal world adversary described above. Particularly, the distribution of $\mathsf{Ideal}_{f,C,B(z)}(x)$ is induced by B's random choices. Note, based on the structure of its inputs, B can perfectly distinguish between outputting x'_C, choosing premature abort, or making the final output. Instead of considering a single algorithm B it would be possible to use, for example, three algorithms B_1, B_2, B_3 corresponding to the three choices. However, this would require to maintain a state between these algorithms which is circumvented by using one algorithm B with a fixed random tape as in Definition 2.18.

B's capabilities are generally inevitable in any real protocol computing f in our model of communication. Clearly, any party P_i can always choose to run the protocol with *substituted inputs* x'_i or to *refuse participation* at all. In the ideal world, refusing participation for some parties $i \in C$ may be indicated using $x'_i = \bot$. It depends on the concrete functionality f whether still meaningful values $y_i \neq \bot$ are sampled if not all parties provide a valid value. Furthermore, any real world protocol in our communication model consists of at least one party which obtains her final output first and, afterwards, can refuse to participate any further. This is due to the impossibility of sending messages only if some other proper message is concurrently being in transmission: It might be known whether a message was sent, but the message's contents remain unknown until it arrives. The refusal of further interaction can be compensated by the honest parties only if $|C| < \lfloor \frac{n}{2} \rfloor$, i. e. the corrupted parties build a strict minority [Gol04]. If we would not allow B to prematurely abort, for any (real world) protocol Π computing f and any $d \geq \lceil \frac{n}{2} \rceil$ there would exist some adversary with $C \subseteq [n]$, $|C| = d$, whose actions could not be simulated in the ideal world. This would imply that no secure protocol for corrupted majorities could exist, because a real world adversary which is not simulatable by any ideal world adversary is considered a security breach by the following definition:

Definition 2.19 (d-Secure MPC)**.** *Let $f\colon (\{0,1\}^*)^n \to (\{0,1\}^*)^n$ be an n-ary functionality and Π be an n party protocol for computing f (Definition 2.17). Protocol $\Pi = (\Pi_1,\dots,\Pi_n)$ is said to* d-securely compute *f if for every ppt A (representing a real model adversary) there exists a ppt B (representing an ideal model adversary), such that for every $C \subseteq [n]$ with $|C| \leq d$*

$$\left\{\mathsf{Ideal}_{f,C,B(z)}(x_C, x_{-C})\right\}_{x_C,x_{-C},z} \overset{\text{comp.}}{\equiv} \left\{\mathsf{Out}^{A,\Pi_{-C}}((x_C,C,z),x_{-C})\right\}_{x_C,x_{-C},z}.$$

*We say Π computes f perfectly d-secure, if for every (*unbounded*) machine A, there exists a machine B such that for all $C \subseteq [n], |C| \leq d$, the distributions above are distributed identically.*

Hence, a protocol is d-secure, if for any real world adversary ppt there exists an ideal world adversary B which, using its limited capabilities, can simulate A's attack on the protocol computationally indistinguishable in the ideal world. In particular this means, that B, essentially,

can simulate A's knowledge, represented by A's output distribution, by an execution within the ideal world. This attack, however, breaks down to substituting x_C by another input x'_C and, after obtaining back the value $f(x'_C, x_{-C})$ aborting the protocol prematurely. As stated above, these are capabilities any real world adversary has with respect to any protocol. Furthermore, the honest parties' output distribution in the ideal world exectution is (almost) the same as when running the real protocol. Hence, whatever the adversary does, the honest parties' outputs are distributed such that either $f(x_{-C}, x'_C)$ for some x'_C is output or $\perp$ is output when A did prematurely abort. Again, no matter what a ppt A does during the execution of Π, she could have done the same by substituting her inputs in the beginning and (possibly) prematurely aborting, when obtaining back f's computed value.

3

Rational Cryptography

In rational cryptography we focus on designing ITM-based protocols which solve cryptographic MPC problems within a rational setting. Therefore, we model the cryptographic problems as games, where the parties inputs are modeled using types, alternative strategies are (possibly restricted) ITMs, and rationality is modeled using utility functions. In this thesis we consider the problems of reconstructing shared secrets as well as the joint evaluation of a function using private inputs within rational settings. Protocols, informally, "solve" these problems, when they almost always efficiently lead to a desired result like all parties correctly reconstructing the secret while no rational party has an incentive to use different ITMs. In game-theoretic terms, we design practical mechanisms (Definition 2.10) where the prescribed ITMs additionally lead to the desired result. These protocols are then considered secure in the sense that no rational party deviates from their prescription due to her own utility. This stands in contrast to traditional MPC security as in Section 2.6 where we analyze protocols in the presence of parties divided into either *arbitrarily malicious* or *completely honest*.

Within this chapter we explicitly model the games and notions used for rational secret sharing (Section 3.2) and rational MPC (Section 3.3). Before this, we define games and corresponding notions modeling computationally bounded players in Section 3.1. These definitions are similar to the game-theoretic framework established in Section 2.4. This is necessary because some cryptographic MPC problems require computational assumptions in order to have *practically implementable* protocols solving them. In the presence of unbounded players or unbounded ITMs such assumptions do not hold. For example, as suggested at the end of Section 2.5, if a secret-sharing is not perfectly d-authenticated, d and more colluding *unbounded* players or ITMs can forge shares to make other parties learn a wrong value while learning the correct secret themselves. Assuming the, arguably, natural incentive to learn secrets while preventing others from doing so, any rational player or coalition could gain from choosing an (unbounded) ITM which computes forged shares. Hence, intuitively, no protocol with the desired property of making all parties eventually learn the correct secret could form a d-resilient equilibrium as long as the secret-sharing scheme is not perfectly d-authenticated.

3.1 Computational Game Theory

In this section we provide a game-theoretic model which adapts the notions from Section 2.4 to cover settings where players are assumed to be computationally bounded. Opposed to the previous game-theoretic notions, following definitions are non-standard. This is because the subject

of rational cryptography is rather young and, so far, there has been no common agreement on (de facto) standard notions. Furthermore, many authors leave details implicit and only give intuitions how the existing game-theoretic notions transfer into a computational model. Here, we distill and formalize notions closely to the definitions and ideas of Katz [Kat08] and Dodis and Rabin [DR07]. We start with defining computational games.

Computational games are always defined with respect to a security parameter $\kappa \in \mathbb{N}$ on which the sampled types and utilities depend. In particular, for every κ and every player P_i there exists a finite set of types $T_i(\kappa)$. Analogue to typed games, the distribution $\mathcal{D}(\kappa)$ then is over $\bigtimes_{i\in[n]} T_i(\kappa)$, i. e. there might exist correlations between sampled types. Besides information related to the concrete game, types are intended for encapsulating information like generated public keys to encrypt communication or validate signatures given to every player. When, for example, reconstructing secrets, types may contain *signed* shares and, additionally, a common public validation key given. Strategies within computational games are ppt ITMs which, depending on the concrete game, might have a certain structure. For example, when jointly evaluating a function with range Y, the final (local) output of any player's machine is required to always be an element $y \in Y$. Because strategy profiles consist of ITMs we may also refer to them as protocols. After the game ends, P_i's utility depends on the concrete sampled types, thus, implicitly the security parameter κ, and, in the most general sense, on anything the players learn during the protocol run, i. e. on the resulting views of all machines. We simplify the latter slightly and make the utilities depend only on the machines' final local outputs instead of their views. Especially when these outputs are restricted to fixed ranges like in secret reconstruction and function evaluation, this allows us to model the player's utilities using simple maps. The (a-priori) utility of a strategy profile $M_{[n]}$ is computed as expectation over sampling the types $t_{[n]}$ and then sampling an output from the interaction of $M_{[n]}$ on $t_{[n]}$. The latter is analogue to sampling a pure strategy from a mixed one in the non-computational setting: Sampling outputs of a protocol corresponds to sampling and fixing the machines' random tapes which makes them and their output deterministic. Conditioned on obtaining type t_i, P_i's (a-posteriori) utility changes to the expectation over $\mathcal{D}(\kappa)^{|t_i}$, i. e. distribution over $\mathcal{D}(\kappa)$ conditioned on t_i was sampled. We formalize these considerations in following definition.

Definition 3.1 (Computational Game)**.** *A* computational game $\hat{\Gamma} = \left(\{S_i\}_{i_\in[n]}, \{u_i\}_{i\in[n]}, \{\mathcal{D}(\kappa)\}_{\kappa\in\mathbb{N}}\right)$ with n players $P_1 \ldots, P_n$ *consists of*

1. *A family* $\{\mathcal{D}(\kappa)\}_{\kappa\in\mathbb{N}}$ *of type distributions* $\mathcal{D}(\kappa) \in \Delta(\mathbb{T}_\kappa)$ *for* $\mathbb{T}_\kappa \coloneqq \bigtimes_{i\in[n]} T_i(\kappa)$ *with finite type sets* $T_i(\kappa)$ *where for all* $t_i \in T_i(\kappa)$ *we have* $|t_i| = \kappa$.

2. *A set of* ppt *machines* $S_i \subseteq$ ITM *for every player* P_i.

3. *A map* u_i *from type vectors (implicitly containing security parameter* $\kappa \in \mathbb{N}$*) and possible* outputs *of* $M_{[n]} \in S_{[n]}$ *on the given type vector to* $\mathbb{R}$ *for every player* P_i. *Formally, let* $\mathbb{T} \coloneqq \bigcup_{\kappa\in\mathbb{N}} \mathbb{T}_\kappa$ *be the set of all types for all* $\kappa \in \mathbb{N}$ *and* $\mathbb{O} \coloneqq \bigcup_{M_{[n]}\in S_{[n]}} \bigcup_{t_{[n]}\in\mathbb{T}} \operatorname{supp}(\mathsf{Out}^{M_{[n]}}(t_{[n]}))$ *be the set of all possible final outputs for any given types and choosable machines. Then,* u_i *is defined as (partial) map* $u_i \colon \mathbb{T} \times \mathbb{O}_i \to \mathbb{R}$.

P_i*'s utility on strategy profile* $(M_1, \ldots, M_n)$ *and security parameter* κ *is*

$$u_i(\kappa, M_{[n]}) \coloneqq \mathbb{E}[t_{[n]} \leftarrow \mathcal{D}(\kappa), o_{[n]} \leftarrow \mathsf{Out}^{M_{[n]}}(t_{[n]}) : u_i(t_{[n]}, o_{[n]})].$$

Given type t_i, P_i*'s utility is denoted by* $u_i(t_i, M_{[n]})$ *and instead of* $\mathcal{D}(\kappa)$ *the expectation is taken over* $D(\kappa)^{|t_i}$, *i. e. distribution* $\mathcal{D}(\kappa)$ *conditioned on type* i *being* t_i.

Now that we have defined computational games, we consider when such strategy profiles or protocols are stable in the presence of *computationally bounded* rational players. When talking about stability we mean d-resilient equilibria where no coalition of size at most d gains by deviating from their prescribed strategy, i. e. ITM. However, different from Section 2.4, in the computational context we allow negligible gains by switching strategies. It is reasonable to assume that parties only care about gains exceeding a certain, fixed lower bound, for example, to compensate invested resources used to find and play alternative strategies. By choosing κ adequately, any negligible gain from using a different strategy falls below this bound and, thus, is irrelevant to the party. Additionally, negligible gains are usually inevitable, since ITMs can always guess solutions for any computationally hard problem provided within a type. If such a problem is solved a party, arguably, gains utility, e. g. by forging new signed shares in secret reconstruction. Hence, a prescribed ITM M_i could always be improved by running an ITM M_i' which first guesses solutions for a given computational problem. Then, if the problem is solved it runs a machine leading to some utility which is $\delta(\kappa)$ higher than running M_i; otherwise M_i' runs M_i. The expected improvement of such a strategy would be $\mu(\kappa) \cdot \delta(\kappa)$ for some negligible function μ. As long as the maximal possible gain from breaking a computational assumption is polynomially bounded, the gain by trying to break a computational assumption is negligible.

Definition 3.2 (Computational d-Resilient Equilibrium)**.** *Let $1 \leq d \leq n$ and computational game $\hat{\Gamma} = \left(\{S_i\}_{i\in[n]}, \{u_i\}_{i\in[n]}, \{\mathcal{D}(\kappa)\}_{\kappa\in\mathbb{N}}\right)$. We call strategy profile $(M_1, \ldots, M_n)$* computational d-resilient equilibrium*, if for all coalitions $C \subseteq [n]$ with $|C| \leq d$, all $i \in C$ and all $M_C' \in \mathrm{S}_C$ we have*

$$u_i(\kappa, M_C', M_{-C}) \lesssim u_i(\kappa, M_{[n]}).$$

In other words, the chosen strategies are in an equilibrium, if the gain of switching strategies is, asymptotically, at most negligible. This is analogue to the cryptographic idea where breaking a security property with negligible chance is not considered as success for adversaries. Different to d-resilient equilibria (Definition 2.6) from Section 2.4, in the computational context it makes a difference that we consider the expected utility of a strategy before and not after obtaining a type. In particular, for any fixed computational problem there exists a solution which can be hardwired within a machine. Therefore, for every type containing computationally hard problems there exists a machine solving these and, arguably, improving the utility compared to machines not solving them. Hence, requiring $u_i(t_i, M_C', M_{-C}) \lesssim u_i(t_i, M_{[n]})$ for all t_i and all M_C' while relying on a computational assumption is generally impossible to instantiate.

Problems of IDoWDS in Computational Settings

As described in Section 2.4, d-resilient equilibria might still have undesirable properties in reality. In particular, if a strategy s_i exists which weakly dominates her equilibrial strategy s_i', a rational party arguably takes the weak domination as tie breaker and chooses s_i' as her strategy. To reflect that strategies within an equilibrium should not be weakly dominated by other strategies, we introduced the notion of IDoWDS (Definition 2.9) and expect equilibria from *practical* mechanisms to survive IDoWDS. However, adapting IDoWDS in a meaningful way to computational games and MPC protocols which rely on computational assumptions is hard or even impossible. Indeed, the first intuition might be to redefine weak domination (Definition 2.8) with respect to computational games and, then, simply apply Definition 2.9. A reasonable definition might consider a strategy, i. e. ITM, as weakly dominated, if another strategy exists which is never more than negligibly worse, but sometimes more than negligibly better. If we assume a polynomially bounded gain for machines which solve the provided computationally hard problem, this property seems meaningful: As long as both run a polynomial number of steps, a machine M is

not weakly dominated by M' which follows the same instructions as M and simultaneously tries to find a solution s_i in the superpolynomial, but finite set of solutions $S(\kappa) = \{s_1, s_2 \ldots, s_r\}$. However, machine M^* which checks in interaction round i whether solution s_i corresponds to the given problem, is guaranteed to find a solution after at most r steps. In the secret reconstruction or MPC context players have no incentive to beneficially interact within or after such a (de facto) last round. Hence machines interacting beneficially in round r or later are weakly dominated and, thus, are deleted by the IDoWDS. Now, $r-1$ becomes the (de facto) last round and by backwards induction only machines which do not interact beneficially at all remain. This problem was noted by Garay et al. [Gar+13] and, to the best of our knowledge, has not been solved satisfactorily yet. Indeed, the problem is inevitable when instantiating private channels with public key encryption and a public key infrastructure: Any polynomial number of encrypted messages can be decrypted with certainty by iterating through the whole key space. Authenticity, for example, when using signed shares, could still be maintained by invalidating old signing keys and reinitializing new ones every constant or polynomial number of rounds. However, we believe to adapt protocols to prevent possibilities which in reality will almost never occur is not the correct approach. Rather the notion needs to be adapted in a meaningful way. In lack of a suitable alternative, we do *not define practical* mechanisms for computational games but only consider the adaption of normal mechanisms.

Definition 3.3 (Computational Mechanism)**.** *For a computational game $\hat{\Gamma}$ and a strategy profile $M_{[n]}$ we call $(\hat{\Gamma}, M_{[n]})$ a* computational d-resilient mechanism*, if $M_{[n]}$ is a computational d-resilient equilibrium in $\hat{\Gamma}$.*

We believe, the problems above can be resolved by using a more complex model for weak domination with respect to computational games. Concretely, the issue is the existence of a *known, fixed* round in which, though being practically never reached within any expected polynomial time protocol, all parties prefer to stop interacting in a beneficial way. However, results reached after a superpolynomial number of interaction rounds should be irrelevant for computationally bounded players. We think, the ideas of Halpern and Pass [HP10] and Maleka, Shareef, and Rangan [MSR08] might be interesting to tackle this problem. Halpern and Pass [HP10] consider costly computation where, with increasing computational resources invested, the gained utilities decrease over time. Maleka, Shareef, and Rangan [MSR08], essentially, model costs using so-called discount factors which shrink the utility on a long-term run. Inspired by their notions, we briefly describe an *idea* which might be analyzed further in subsequent works. Particularly, we think of sampling a *personal* last round r_i for every player P_i together with her type. If P_i does not choose its output before round r_i ends, she obtains 0 utility. This can be seen as model for cases where P_i has set herself a fixed limit of time and other resources she is willing to spend. These values r_i might be chosen, such that (1) the inspected protocol ends with high probability within less rounds, (2) computational problems cannot be efficiently solved within the number of rounds, and (3) every party has a low probability of predicting another party's final round. Under these conditions, P_i's machines interacting after round r_i are even strictly dominated by machines which do not and, thus, are deleted by the IDoWDS. Hence, P_i's remaining strategies considered by the IDoWDS are these which interact for at most r_i rounds. Because these rounds r_i are no common knowledge, the backwards induction argument, which relies on the property that every party knows, when a final round is reached does not apply anymore. As noted above, this is only an idea which could not be analyzed further within this thesis.

3.2 Rational Secret Sharing

In rational secret sharing we consider the problem of reconstructing shared secrets in the presence of rational players. Reconstructing secrets is a very natural task within secure MPC protocols satisfying Definition 2.19. Indeed, in the sense of Goldreich [Gol04, Definition 7.3.13], MPC protocols are canonical if they proceed in two phases: First, run a protocol which securely computes a sharing of the evaluated functionality's value and distributes the shares among the parties. Second, the parties reconstruct the secret using some given protocol. Halpern and Teague [HT04] were the first to notice and prove the impossibility of any reconstruction protocols with a fixed upper bound on their round length in the rational setting. Before elaborating on their arguments we explicitly model the problem of reconstructing shared secrets as a game.

Similar to Asharov and Lindell [AL11], we define the setting for players which possibly obtain auxiliary information related to the shared secret. Comparable to the traditional MPC framework (c. f. Section 2.6), providing auxiliary information is often necessary when we need to preserve a protocol's properties while being used as subcomponent of another protocol. The secret reconstruction game is played by n players with respect to some secret-sharing scheme $(\mathsf{Share}, \mathsf{Recon})$ having domain of secrets Y. In the game's beginning, some secret and corresponding auxiliary information are chosen according to a publicly known distribution. Then a sharing is computed and every player is given a type containing her auxiliary information and share. We assume the players may only pick an ITM which eventually outputs a value $y \in Y$. Like many other works in the field of rational secret-sharing (e. g. [HT04; GK06; Ong+09; FKN10; LS10]), we assume a player's utility to depend only on which secret was chosen and which guesses were eventually made by the players. Note, this model simplifies reality, for example, due to ignoring the players' confidences in their guesses: For example, with respect to future actions a player may gains more when another party puts all her trust in some falsely reconstructed value instead of being more cautious. Such considerations were first taken closer into account by Asharov and Lindell [AL11] and, based on their results, by De and Pal [DP13].

Definition 3.4 (Secret Reconstruction Game)**.** *The* secret reconstruction game $\Gamma_{S,\mathcal{D},u}$ *(with auxiliary inputs) is a typed game with secret-sharing scheme* $S = (\mathsf{Share}, \mathsf{Recon})$ *having domain of secrets* Y*, secret and auxiliary input distribution* $\mathcal{D} \in \Delta(Y \times (\{0,1\}^*)^n)$ *and utility maps* $u_i \colon Y \times Y^n \to \mathbb{R}$*. P_i's strategies are all ITMs M_i which output a guess $y_i \in Y$ after their interaction ends. In the game, first, secret y and auxiliary inputs $a_{[n]}$ are chosen, i. e. $(y, a_{[n]}) \leftarrow \mathcal{D}$. Then a sharing of y is generated by $s_{[n]} \leftarrow \mathsf{Share}(y)$ and every player P_i privately obtains her type (s_i, a_i). For strategy profile $(M_1, \ldots, M_n)$, P_i's utility is*

$$u_i(M_{[n]}) \coloneqq \mathbb{E}[(y, a_{[n]}) \leftarrow \mathcal{D}, s_{[n]} \leftarrow \mathsf{Share}(y), y_{[n]} \leftarrow \mathsf{Out}^{M_{[n]}}((s,a)_{[n]}) : u_i(y, y_{[n]})].$$

In the computational version*, denoted by $\hat{\Gamma}_{S,\mathcal{D},u}$, there is a security parameter $\kappa \in \mathbb{N}$ such that $\mathsf{Share}(1^\kappa, y)$ computes the shares (of size κ), the types may contain additional inputs (i. e. generated computationally hard problems) and strategies are ppt computable on their non-auxiliary input.*

The desired goal for such a reconstruction setting is a protocol which makes *every* party learn the shared secret *correctly* as well as *efficiently*. Additionally to these properties, it needs to be stable with respect to the players' incentives, i. e. coalitions of up to d players must not gain (noticeably) by deviating from the protocol. In game-theoretic terms, we design a (possibly computational) d-resilient practical mechanism for $\Gamma_{\mathcal{D},S,u}$ which has the desired goals stated above.

Definition 3.5 (Secret Reconstruction Mechanism)**.** *Let $\Gamma_{S,\mathcal{D},u}$ be a secret reconstruction game (Definition 3.4). A strategy profile $M_{[n]}$ is called* d-resilient secret reconstruction mechanism *for $\Gamma_{S,\mathcal{D},u}$, if the following conditions hold:*

1. *For all* $(y, a_{[n]}) \in \mathrm{supp}(\mathcal{D})$, $s_{[n]} \in \mathrm{supp}(\mathsf{Share}(y))$ *and* $0 < \varepsilon \leq 1$ *there exists a polynomial* $p\colon \mathbb{R} \to \mathbb{N}$ *such that* $\Pr[\mathsf{Out}^{M_{[n]}}((s,a)_{[n]}) = y^n] \geq 1-\varepsilon$ *and, additionally,* $\mathsf{Time}^{M_{[n]}}((s,a)_{[n]}) \leq p(\varepsilon)$, *i. e. the expected runtime is polynomially bounded in* ε.

2. $M_{[n]}$ *is a d-resilient equilibrium in* $\Gamma_{S,\mathcal{D},u}$, *i. e.* $(\Gamma_{S,\mathcal{D},u}, M_{[n]})$ *is a d-resilient mechanism.*

If $M_{[n]}$ *additionally survives IDoWDS (Definition 2.9), it is further called practical. For the computational game* $\hat{\Gamma}_{S,\mathcal{D},u}$ *we, instead of conditions 1 and 2, require correctness with respect to a negligible function* μ, *and a* computational *d-resilient mechanism (Definition 3.2).*

Although not modeling all these details, Halpern and Teague [HT04] essentially show, that no protocol with a known final roundcan be a practical reconstruction mechanism with respect to reasonable assumptions on the player's utilities. Concretely, the compare two resulting outputs from a protocol run: On the one hand, P_i's utility is *strictly higher* when she correctly outputs the shared secret compared to when she does not. On the other hand, when another party outputs a wrong value instead of the correct one, P_i's utility does never decrease and, additionally, for at least one party even increases. Dodis and Rabin [DR07] refer to these properties as *correctness* and, respectively, *exclusivity*. Utility functions reflecting these properties are called *natural* for secret reconstruction by Asharov and Lindell [AL11]. Different to Asharov and Lindell [AL11], utilities in our scenario may depend on the concrete guesses for the secret and not just a binary vector indicating right and wrong guesses.

Definition 3.6 (Natural Reconstruction Utilities)**.** *We call functions* $u_{[n]}$ *with* $u_i\colon Y \times Y^n \to \mathbb{R}$ natural reconstruction utilities *if for all* $i \in [n]$, $y \in Y$ *and* $y_{[n]}, y'_{[n]} \in Y^n$ *we have*

1. *(Correctness)* $u_i(y, y_{[n]}) > u_i(y, y'_{[n]})$, *if* $y_i = y \neq y'_i$.

2. *(Exclusivity)* $u_i(y, y_{[n]}) \geq u_i(y, y'_{[n]})$, *if* $\exists j^* \in [n]$ *with* $y_{j^*} = y \neq y'_{j^*}$ *and for all* $j \neq j^*$ *we have* $y_j = y'_j$.

Note, if too many players do not gain *strictly* from exclusivity, i. e. have no "rivals", the trivial protocol instructing every party to broadcast her shares is d-resilient. If this is not the case and if we assume natural reconstruction utilities, the impossibility of reconstruction mechanisms with deterministic last round is due to the iterated deletion of weakly dominated strategies (Definition 2.9). Intuitively, for any player P_i with rivals, revealing "beneficial" information in the last round is weakly dominated by not sending anything and, thus, deleted as selectable strategy. To see this, recall a strategy M_i being weakly dominated by M'_i means, that M_i never yields higher, but sometimes yields lower utility than M'_i. Revealing beneficial information in the last round does not affect the own knowledge and, thus, not the own output's correctness. Indeed, it might only increase the others' probabilities of correctly outputting the secret. Hence, by exclusivity and correctness, the utility cannot be higher than when not revealing the information. Moreover, it is possible to construct machines M_{-i} such that at least one rival's probability to correctly guess the secret is increased by the revealed information. Thus, all ITMs M_i which reveal new information in the last round are deleted by the process of IDoWDS. This makes the second-last round the effective last round and the process inductively applies until no ITM revealing beneficialinformation remains.

Besides their impossibility theorem, Halpern and Teague [HT04] prove the existence of a protocol with indefinite last round which, essentially, builds a 1-resilient practical secret reconstruction mechanism assuming a (perfectly) n-authenticated secret-sharing scheme, more than two players, and simultaneous communication channels. Gordon and Katz [GK06] and Abraham et al. [Abr+06] enhance this result and cover settings with two and more players and higher d-resiliency while relaxing the secret-sharing scheme to be only (perfectly) d-authenticated. Their

ideas have in common, to introduce an uncertainty about whether the current interaction round is the effectively last one, i. e. reveals the secret. Furthermore, if a party deviates from the protocol this is caught immediately after an interaction round and makes all parties stop further interaction. The uncertainty whether a round reveals the secret is chosen such that the probability of not learning the secret due to being caught outweighs the possible gain of more exclusivity.

In the aforementioned works, it suffices to suppose natural reconstruction utilities in order to instantiate a reconstruction mechanism based on the concrete utility values. However, they, for example, do not model auxiliary information or the choice of secrets at all. Depending on the concrete utility values and distribution over secrets and auxiliary information, guessing the secret instead of running any protocol might be favorable to rational parties. For instance, assume there are two parties P_1, P_2 and secrets are chosen uniformly at random from $Y = \{y_1, y_2\}$ without auxiliary information. Solely based on her share, any party guesses the secret correctly with probability $\frac{1}{2}$ by the sharing scheme's perfect privacy. Now, suppose P_1's utility is 10 when she outputs the correct secret while P_2 is wrong, 3 when both output the correct secret, 2 when both are wrong, and 1 when she is wrong while P_2 is right. The ITM which chooses its output y uniformly at random from Y without any interaction leads, independently of P_2's strategy, to an expected utility of $\frac{1}{4}(10 + 3 + 2 + 1) = 4$ for P_1. Any protocol which (almost always) leads both parties to the correct result yields the strictly lower expected utility of 3. Hence, even assuming a (perfectly) d-authenticated secret-sharing scheme, having natural utilities is not a sufficient condition for the existence of reconstruction mechanisms in our modeling. To overcome this we define the notion of d-non-cooperatively reconstrutability which, especially, ensures, that guessing secrets does not outperform protocols leading to correct outputs. Therefore, we use a *mediated* typed game (Definition 2.11), i. e. a game where players send (possibly substituted) types to a mediator M and, based on these types, receive back a recommended strategy which they use to choose their final strategy.

Definition 3.7 (d-Non-Cooperatively Reconstructable)**.** *The* ideal secret reconstruction game $\Gamma^{ideal}_{S,\mathcal{D},u,\mathcal{M}}$ *corresponding to a secret reconstruction game* $\Gamma_{S,\mathcal{D},u}$ *(Definition 3.4) is a* mediated *typed game (Definition 2.11) where every player P_i's (final) strategy set is equal to the domain of secrets Y and the mediator strategy $\mathcal{M}$ is*

$$\mathcal{M}((s,a)_{[n]}) := \begin{cases} \mathsf{Recon}(s_{[n]})^n, & \text{if } \mathsf{Recon}(s_{[n]}) \neq \perp \\ y_{[n]} \text{ with } y_i \leftarrow \arg\max_{y \in Y} \Pr[\mathcal{D}^{|a_i} = (y, \cdot)], & \text{otherwise.} \end{cases}$$

In the game, first, secret and auxiliary information $(y, a_{[n]}) \leftarrow \mathcal{D}$ and shares $s_{[n]} \leftarrow \mathsf{Share}(y)$ are sampled. Every player P_i then obtains $(s,a)_i$, samples $(s', a')_i \leftarrow \delta^1_i((s,a)_i)$ and sends it to the mediator. Mediator M samples $y_{[n]} \leftarrow \mathcal{M}((s',a')_{[n]})$, returns y_i back to P_i who samples her final strategy $y'_i \leftarrow \delta^2_i((s,a)_i, (s',a')_i, y_i)$. Secrets are d-non-cooperatively reconstructable *in $\Gamma_{S,\mathcal{D},u}$, if playing the canonical strategy, i. e. truthfully inputting $(s,a)_i$ into $\mathcal{M}$ and outputting the recommendation, is a d-resilient equilibrium in $\Gamma^{ideal}_{S,\mathcal{D},u,\mathcal{M}}$.*

Secrets are computationally d-non-cooperatively reconstructable*, if playing the canonical strategy is a computational d-resilient equilibrium for the computational version of $\Gamma^{ideal}_{S,\mathcal{D},u,\mathcal{M}}$.*

In other words, unless Recon detects invalid shares, the mediator reconstructs a secret using shares from the (possibly substituted) types. If Recon detects invalid shares, M computes and recommends to every player the likeliest secret with respect to $\mathcal{D}$ conditioned on the provided auxiliary information a_i. If sending true types and outputting the recommendation builds a d-resilient equilibrium, we say secrets are d-non-cooperatively reconstructable (d-NCR). Within our setting, d-NCR is a necessary condition for the existence of a (practical) d-resilient reconstruction mechanism. If d-NCR does not hold, then no correct reconstruction protocol can be

d-resilient in the particular setting. Concretely, there exists a size-d coalition which gains from replacing her (computationally) non-d-authenticated shares or aborting the protocol and guessing the secret. Since we will always assume a (computationally) d-authenticated sharing, d-NCR effectively prevents that guessing secrets is a better alternative to interaction. If a d-resilient reconstruction mechanism preserves its properties within any setting which is d-NCR, we call it *utility independent* in our model. Our notion is similar to utility independence of Asharov and Lindell [AL11] who, as noted above, take slightly different utility functions into account.

Definition 3.8 (Utility Independent Reconstruction Mechanism)**.** *A d-resilient secret reconstruction mechanism $M_{[n]}$ (for* some *secret reconstruction game $\Gamma_{S,\mathcal{D},u}$) is called* utility independent*, if $M_{[n]}$ is a d-resilient secret reconstruction mechanism for* any *secret reconstruction game $\Gamma_{S,\mathcal{D}',u'}$ (Definition 3.5) where utilities u' and distribution $\mathcal{D}'$ are such that secrets are d-non-cooperatively reconstructable (Definition 3.7).*

Analogously, utility independence is defined for a practical (resp., computational) d-resilient reconstruction mechanism $M_{[n]}$, i. e. $M_{[n]}$ should remain a practical (resp., computational) mechanism within any d-NCR (resp., computational d-NCR) setting. Utility independence is required when instantiating protocols without knowledge of the participants' concrete utilities. This is relevant as in reality even parties themselves are incapable of stating their explicit utility values. Additionally, utility independence is also an advantageous property when using a reconstruction mechanism as another protocol's subcomponent as we will do. Particularly, as long as the outer protocol guarantees a reconstruction setting which is d-NCR, all parties will stick to their prescription even if d of them collude. Our assumptions and the constructed protocol for rational MPC or, more precise, rational function evaluation ensure a d-NCR environment when the secret reconstruction mechanism is run as subcomponent. Utility independence yields that no size-d coalition has a better alternative than sticking to the reconstruction protocol.

3.3 Rational MPC and Function Evaluation

MPC focuses, as described in Section 2.6, on the problem of jointly computing a given functionality f's outputs correctly while preserving the privacy of the used inputs. In rational MPC we analyze protocols in the presence of rational players whose interests, primarily, lie in learning the correct output while, secondarily, e. g. keeping their own inputs private, learning further information on the other players' inputs or preventing other players from gaining information.

In their work on rational secret sharing, Halpern and Teague [HT04] noted that traditionally secure MPC protocols (c. f. Section 2.6) face the same problems as secret reconstruction protocols with respect to rational players. Indeed, their argument on the non-existence of a secret reconstruction mechanism with deterministic number of rounds applies to protocols computing f as well: With respect to not sending anything in the last round, revealing beneficial information is weakly dominated and, thus, deleted by the IDoWDS Hence, the second-last round becomes the effective last round and, by backwards induction, their statement then follows (c. f. discussion after Definition 3.6).

In this thesis, we consider the problem of function evaluation, which is a slightly less general problem of rational MPC than Halpern and Teague [HT04] consider. We focus on the *evaluation of single-output functions* $f\colon X_1 \times \cdots \times X_n \to Y$ where the only information of interest to the players is f's output for the provided inputs $x_1, \ldots, x_n$. Indeed, we formalize the ideas described by Gordon and Katz [GK06, Section 5] within our game-theoretic framework. Following structure and notions are mostly analogue to Section 3.2. Therefore, explanations and discussions which are similar to those from rational secret sharing are less comprehensive. First, we define a game which models function evaluation for rational players in reality. As in secret

reconstruction, the utilities solely depend on the final guesses on the values which is a simplification. Considering goals from MPC, this simplification is rougher than in secret reconstruction because, for example, knowledge beyond the function value gained when playing the game is not reflected in the utilities. Note, our model could be extended to cover this by removing the ITMs' restrictions to output a single value from Y and allowing to output additional values related to $X_{[n]}$ or Y. More general, we could also consider the parties' views after the protocol execution. However, we consider the more basic model in order to provide clean formal proofs which, in future, can be extended formally to more complex models.

Definition 3.9 (Function Evaluation Game (Real))**.** *The* (real) function evaluation game $\Gamma_{f,\mathcal{D},u}$ *is a typed game for function* $f\colon X_{[n]} \to Y$ *with finite sets* $X_1, \ldots, X_n, Y$*, type distribution* $\mathcal{D} \in \Delta(X_{[n]})$ *and utilities defined by* $u_i\colon Y \times Y^n \to \mathbb{R}$*. The players' strategies are ITMs which output an element from* Y *after their interaction on the sampled types ends. Formally, for strategy profile* $(M_1, \ldots, M_n)$ P_i*'s utility is*

$$u_i(M_{[n]}) = \mathbb{E}[x_{[n]} \leftarrow \mathcal{D}, y_{[n]} \leftarrow \mathsf{Out}^{M_{[n]}}(x_{[n]}) : u_i(f(x_{[n]}), y_{[n]})].$$

In the computational version, $\hat{\Gamma}_{f,\mathcal{D},u}$*, there is a security parameter* $\kappa \in \mathbb{N}$ *such that for sufficiently large* κ *we have* $|x_i| = \kappa$*, the types may contain additional inputs (e. g. generated computational hard problems), and strategies are ppt computable on their input.*

Note, differently to the secret reconstruction game, the players obtain no explicit auxiliary information alongside their inputs $x_{[n]}$ sampled according to $\mathcal{D}$. However, because $\mathcal{D}$ is a distribution over $\bigtimes_{i\in[n]} X_i$, the values might be correlated and therefore possibly carry implicit auxiliary information. For rational function evaluation the goal is to design protocols which lead all parties to (almost always) output the correct function value with respect to the sampled types. As in secret reconstruction, following the protocol is required to be a d-resilient equilibrium, and the mechanism is called practical if it additionally survives IDoWDS. Furthermore, natural utilities do reflect the incentive to, primarily, learn the correct function value for the given values $x_{[n]}$ while, secondarily, preventing other parties from learning the value.

Definition 3.10 (Function Evaluation Mechanism)**.** *Let function evaluation game* $\Gamma_{f,\mathcal{D},u}$ *(Definition 3.9). A strategy profile* $M_{[n]}$ *is called* d-resilient function evaluation mechanism *for* $\Gamma_{f,\mathcal{D},u}$*, if the following conditions hold:*

1. *For any* $x_{[n]} \in \mathrm{supp}(\mathcal{D})$ *and* $0 < \varepsilon \leq 1$ *there exists a polynomial* $p\colon \mathbb{R} \to \mathbb{N}$ *such that* $\Pr[\mathsf{Out}^M(x_{[n]}) = (f(x_{[n]})^n)] \geq 1 - \varepsilon$ *and* $\mathsf{Time}^M(x_{[n]}) \leq p(\varepsilon)$*, i. e. the expected runtime is polynomially bounded in* ε*.*

2. $M_{[n]}$ *is a* d*-resilient equilibrium in* $(\Gamma_{f,\mathcal{D},u})$*, i. e.* $(\Gamma_{f,\mathcal{D},u}, M_{[n]})$ *is a* d*-resilient mechanism.*

If $M_{[n]}$ *additionally survives IDoWDS (Definition 2.9), it is further called practical. For the computational game* $\hat{\Gamma}_{f,\mathcal{D},u}$ *we, instead of conditions 1 and 2, require correctness with respect to a negligible function* μ*, and a* computational d*-resilient mechanism (Definition 3.2).*

Definition 3.11 (Natural MPC Utilities)**.** *We call functions* $u_{[n]}$ *with* $u_i\colon Y \times Y^n \to \mathbb{R}$ natural MPC utilities *if for all* $i \in [n]$*,* $y \in Y$ *and* $y_{[n]}, y'_{[n]} \in Y^n$ *we have*

1. *(Correctness)* $u_i(y, y_{[n]}) > u_i(y, y'_{[n]})$*, if* $y_i = y \neq y'_i$*.*

2. *(Exclusivity)* $u_i(y, y_{[n]}) \geq u_i(y, y'_{[n]})$*, if* $\exists j^* \in [n]$ *with* $y_{j^*} = y \neq y'_{j^*}$ *and for all* $j \neq j^*$ *we have* $y_j = y'_j$*.*

As for secret reconstruction, in some settings a function evaluation mechanism may not exist, although we have natural MPC utilities. To characterize settings where an evaluation may exist, we use the property of d-non-cooperatively computable (d-NCC) settings, analogue to d-NCR for secret reconstruction (Definition 3.7). NCC was first introduced by Shoham and Tennenholtz [ST05]. We model the property within our framework closely to the description given in [Kat08]. Therefore, as in Definition 3.7, we define a mediated typed game which is called the ideal function evaluation game. Within this game, the mediator acts as trusted party evaluating the function f for the players' provided (and possibly substituted) inputs.

Definition 3.12 (Function Evaluation Game (Ideal))**.** *The* ideal function evaluation game for f on input distribution $\mathcal{D}$ *is a* mediated *typed game (Definition 2.11)* $\Gamma^{ideal}_{f,\mathcal{D},u,\mathcal{M}}$ *where* $f, \mathcal{D}$ *and* u *are as in the real game (Definition 3.9), every player* P_i*'s (final) strategy set is* Y *and the mediator strategy* $\mathcal{M}$ *is*

$$\mathcal{M}(x_{[n]}) \coloneqq \begin{cases} (f(x_{[n]}))^n, & \text{if } \forall i \in [n] : x_i \neq \perp \\ y_{[n]} \text{ with } y_i \leftarrow \arg\max_{y \in Y} \Pr[x_{[n]} \leftarrow \mathcal{D}^{|x_i} : f(x_{[n]}) = y], & \text{otherwise.} \end{cases}$$

The game is played by sampling types $x_{[n]} \leftarrow \mathcal{D}$ *and sending them to the corresponding players. Every player* P_i *then samples a type* $x_i \leftarrow \delta_i^1(x_i) \in \Delta(X_i \cup \{\perp\})$ *and sends it to the mediator. Mediator* M *samples* $y_{[n]} \leftarrow \mathcal{M}(x'_{[n]})$*, returns* y_i *back to* P_i *who samples her final strategy* $y'_i \leftarrow \delta_i^2(x_i, x'_i, y_i)$*. For given profile* $\delta_{[n]}$*,* P_i*'s utility is*

$$u_i(\delta_{[n]}) = \mathbb{E}[x_{[n]} \leftarrow \mathcal{D}, x'_{[n]} \leftarrow \delta^1_{[n]}(x_{[n]}), y_{[n]} \leftarrow \mathcal{M}(x'_{[n]}), y'_{[n]} \leftarrow \delta^2_{[n]}((x, x', y)_{[n]}) : u_i(f(x_{[n]}), y'_{[n]})].$$

Opposed to the traditional ideal world for MPC (Definition 2.18) we may not allow so-called premature abort where the aborting party (or coalition) learns the function value and the remaining parties obtain a failure symbol $\perp$. Otherwise, early abort would (in most scenarios) display a dominating strategy in the rational model. Abortion is modeled by allowing the parties to provide $\perp$ to M. If at least one input was $\perp$, M advises each player P_i to output the most likeliest function value with respect to distribution $\mathcal{D}$ conditioned on her provided type x_i. Similar to the traditional ideal world model, it could be possible to consider settings where the function is evaluated nonetheless, for example, using a default value for every party. If aborting or substituting inputs is not preferred over using the mediator, the particular setting is called d-NCC.

Definition 3.13 (d-NCC Function)**.** *Let ideal function evaluation game* $\Gamma^{ideal}_{f,\mathcal{D},u,\mathcal{M}}$ *(Definition 3.12). Function* f *is* d-non-cooperatively computable (NCC) *with respect to distribution* $\mathcal{D}$ *and utilities* $u_{[n]}$ *if sending the true type to mediator* M *and outputting the returned value* y*, i. e. playing the canonical strategy (Definition 2.12), forms a* d*-resilient equilibrium (Definition 2.7) in* $\Gamma^{ideal}_{f,\mathcal{D},u,\mathcal{M}}$*.*

Note, different to secret reconstruction, we split d-NCC and the ideal game into two definitions, because we need to refer several times explicitly to one or the other. Furthermore, because every set X_i is finite, the strategies/ITMs in use may have a different behavior for any given x_i encoded. Hence, if for at least one x_i from $\mathcal{D}$'s support there exists a better strategy than providing x_i truthfully to mediator M, then the function cannot be d-NCC. Indeed, based on this consideration, Dodis and Rabin [DR07, Section 1.5.1] provide a concrete classification of functions which may never be part of a d-NCC setting. One such class are dominated functions, i. e. functions where at least one party P_i for some obtained value x_i can determine f's output. Hence, given x_i, P_i does not need mediator M to compute the secret, but can use any substituted x'_i leading the remaining parties to a (possibly) wrong output $f(x'_i, x_{-i})$. The second class of functions is called reversible and, essentially, covers functions where some P_i can input a value

$x_i' \neq x_i$, such that, given $y' = f(x_i', x_{-i})$, she can always compute the correct result $f(x_{[n]})$ while the other parties sometimes do not learn it. For example, $f(x_{[n]}) \coloneqq \sum_{i \in [n]} x_i$ is reversible: P_1 may insert any $x_1' \neq x_1$ with $x_1' \in X_1$ which makes the mediator's recommendation y' wrong for all other players while she can compute the correct value $y' - x_1' + x_1$. An example for dominated functions are max and min where one party given the highest or, respectively, lowest possible value from $\mathcal{D}$ can always determine the output. As long as no other party obtained the same maximal or, respectively, minimal value, she can send a different value to M and, thus, make all other parties learn a wrong value. However, in the realistic case when $\mathcal{D}$ is not perfectly known, both functions might be considered as d-NCC candidates. The kth order statistics, i. e. the kth smallest value of a sampled set, build d-NCC functions for every $d < k < n - d$ assuming pure guessing leads to a worse expected utility than all parties learning the correct value. Assuming the latter, Dodis and Rabin [DR07, Theorem 1.8] state that a function is 1-NCC if and only if it is neither reversible nor dominated. By generalizing the notions of domination and reversibility to coalitions of size d, the analogue is claimed to hold for d-NCC.

4 Function Evaluation Mechanism — Construction and Proof

In this chapter we provide a generic construction for the rational MPC problem of function evaluation. We prove its required (security) properties for the model of function evaluation established in Section 3.3. Particularly, our construction, depending on the assumptions yields, provably, either a d-resilient mechanism for function evaluation or a computational one (c. f. Definition 3.10).

4.1 Generic Construction of a Function Evaluation Mechanism

Before describing the construction itself, we briefly recall the setting of the computational function evaluation game $\hat{\Gamma}_{f,\mathcal{D},u}$ (Definition 3.9) used throughout this section. At the beginning of game $\hat{\Gamma}_{f,\mathcal{D},u}$ with security parameter $\kappa \in \mathbb{N}$ every player P_i chooses a ppt ITM M_i which, after interacting, outputs a value within the range Y of function $f\colon X_{[n]} \to Y$. Then every player P_i obtains a type t_i (with $|t_i| \geq \kappa$) containing $x_i \in X_i$ and (possibly) additional data describing computational hard problems. The values x_i are sampled according to the publicly known distribution $\mathcal{D} \in \Delta(X_{[n]})$. Next, every P_i runs $M_i(t_i)$ which after its interaction with the other machines outputs a value $y_i \in Y$. Player P_i's utility u_i then only depends on the true function value $f(x_{[n]})$ and the outputs $y_{[n]}$, i. e. is defined via $u_i\colon Y \times Y^n \to \mathbb{R}$. We assume natural MPC utilities $u = (u_1, \ldots, u_n)$ which models each player's incentive to, primarily, output the correct value and, secondarily, minimize the number of other players being correct (c. f. Definition 3.11).

Our construction is, essentially, a generalized version of the proposal from Gordon and Katz [GK06, Section5]. First, the parties compute a secret-sharing of $f(x_{[n]})$ using a traditional MPC protocol. Afterwards, they run a secret reconstruction mechanism to obtain this value back again. Remember, it does not suffice to compute $f(x_{[n]})$ directly using traditional MPC protocols as in most settings adversaries can prematurely abort, i. e. learn the value while preventing the others from doing so. We mainly deviate from Gordon and Katz [GK06] by modularizing their approach. Particularly, in the construction we require only *some* secret-sharing scheme and *some* corresponding reconstruction mechanism instead of fixed instantiations. To prove its function evaluation mechanism's properties we make further assumptions on these components but still do not fix concrete schemes. This approach is intended to allow for more flexible implementations of protocols used for rational function evaluation. To facilitate this modularization, besides $\hat{\Gamma}_{f,\mathcal{D},u}$, the number of players n, and the resiliency d we want to achieve, let

1. $S = (\mathsf{Share}, \mathsf{Recon})$ be some (k, n)-secret-sharing scheme (Definition 2.14) with domain of secrets Y for $n > k \geq d$, i. e. coalitions of up to size d, which we will consider, are not able

to reconstruct the secrets without obtaining at least one other player's share.

2. $\Pi = (\Pi_1, \ldots, \Pi_n)$ be some n-party protocol computing the following functionality:

$$g : ((x, \kappa)_{[n]}) \mapsto \begin{cases} \mathsf{Share}(1^{\kappa_1}, f(x_{[n]})) & \text{if } \forall i, j : \kappa_i = \kappa_j \wedge x_i \in X_i \\ \perp & \text{otherwise} \end{cases} \tag{4.1}$$

Note, $(x, \kappa)_{[n]}$ is our short notation for $((x_i, \kappa_i)_{i \in [n]})$ and the security parameter $\kappa \in \mathbb{N}$ is an explicit input of Functionality 4.1. If not all parties insert the same security parameter κ_i, $\perp$ is output by definition. Since we will assume at least one player P_i to insert $\kappa_i = \kappa$ honestly, the remaining rational players have an incentive to do the same. Furthermore, note that Π computing some functionality provides no security guarantees, but only ensures correctness (c. f. Definition 2.17).

3. $M = (M_1, \ldots, M_n)$ be some strategy profile for the computational secret reconstruction game $\hat{\Gamma}_{S,\mathcal{D}',u}$ (Definition 3.4) where $\mathcal{D}'$ is defined by $\Pr[\mathcal{D}' = (f(x_{[n]}), x_{[n]})] \coloneqq \Pr[\mathcal{D} = x_{[n]}]$ for any $x_{[n]} \in X_{[n]}$. Hence, the shared value is sampled according to $f(\mathcal{D})$ and every P_i's auxiliary information is a corresponding x_i. Thus, when P_i obtains x_i alongside her share she can deduce that the shared value was sampled from $f(\mathcal{D}^{|x_i})$ where $\mathcal{D}^{|x_i}$ is $\mathcal{D}$ conditioned on x_i was sampled. The utilities u_i within $\hat{\Gamma}_{S,\mathcal{D}',u}$ are the same as in the function evaluation game $\hat{\Gamma}_{f,\mathcal{D},u}$. Hence, if the value $y = f(x_{[n]})$ was shared and the players' guesses after the interaction are $y_{[n]}$, P_i obtains utility $u_i(y, y_{[n]})$. This is equal to P_i's utility within the function evaluation game when the players output $y_{[n]}$ for the given types $x_{[n]}$.

Construction 4.1 (Generic Function Evaluation Protocol)**.** *Let security parameter $\kappa \in \mathbb{N}$. Further, let $\hat{\Gamma}_{f,\mathcal{D},u}$, $(\mathsf{Share}, \mathsf{Recon})$, $\Pi = (\Pi_1, \ldots, \Pi_n)$, $M = (M_1, \ldots, M_n)$ as described above. On common input 1^κ and private input $x_i \in X_i$ strategy M_i^* is:*

1. *Run $\Pi_i(1^\kappa, x_i)$.*

2. *Let s_i denote the (local) output after the interaction of step 1. If $s_i \neq \perp$, run strategy $M_i(1^\kappa, s_i, x_i)$. Else, stop sending (but continue listening) and output a likeliest correct guess on $f(x_{[n]})$ with respect to $\mathcal{D}$ conditioned on x_i, i. e. $y_i \leftarrow \arg\max_{y \in Y} \Pr[f(\mathcal{D}^{|x_i}) = y]$.*

As mentioned above, Construction 4.1 results in a strategy profile $M^* = (M_1^*, \ldots, M_n^*)$ for function evaluation. Suppose all parties stick to their prescription. Then, in step 1, a secret-sharing of $f(x_{[n]})$ is sampled and distributed among the parties using protocol Π which computes Functionality 4.1. Afterwards, running the strategy profile M for secret reconstruction outputs *some* value from Y after the interaction. However, until now neither correctness nor d-resiliency are included. Following theorem states that M^*, under certain assumptions regarding its components, is a computational d-resilient function evaluation mechanism, i. e. a protocol which almost always makes every player output the correct function value even if up to d rational parties are considered to collude and deviate.

Theorem 4.2. *The strategy profile $M^* = (M_1^*, \ldots, M_n^*)$ from Construction 4.1 is a* computational *d-resilient mechanism for function evaluation in $\Gamma_{f,\mathcal{D},u}$ (Definition 3.10), if following conditions hold:*

(V1) *Function* f *is d-NCC (Definition 3.13) with respect to distribution $\mathcal{D}$ and utilities u.*

(V2) *MPC protocol Π computes Functionality 4.1 d-securely (Definition 2.19).*

(V3) *The strategy profile $M = (M_1, \ldots, M_n)$ is a computational d-resilient secret reconstruction mechanism (Definition 3.5) for $\hat{\Gamma}_{S,\mathcal{D}',u}$ where secret and auxiliary input distribution $\mathcal{D}'$ is defined as $\Pr[\mathcal{D}' = (y, x_{[n]})] \coloneqq \Pr[\mathcal{D} = x_{[n]}]$ for all $x_{[n]} \in X_{[n]}$ and $y \coloneqq f(x_{[n]})$.*

(V4) *The functions $u = (u_1, \ldots, u_n)$ are polynomially bounded.*

(V5) *The reconstruction mechanism M is utility independent (Definition 3.8).*

Note, although Assumption (V5) implies Assumption (V3) we make use of both assumptions. This is due to the fact, that a utility independent reconstruction mechanism is known to be impossible in certain settings. Furthermore, we conjecture that Assumption (V5) is not a necessary condition and weaker properties might suffice to instantiate Construction 4.1 securely. Yet, we were not able to identify weaker sufficient notions existing reconstruction mechanisms have in common. Nonetheless, we chose to explicitly distinguish between Assumptions (V3) and (V5) to facilitate replacing the latter assumption within subsequent works.

Before heading to further results and Theorem 4.2's formal proof, we explain informally how the assumptions lead to a d-resilient function evaluation mechanism. Assumption (V1) is a *necessary* condition for the existence of such a mechanism. Particularly, if a setting is not d-NCC, there exists at least one coalition of size d which has no interest in running any protocol honestly using its true inputs. Next, Assumption (V2) assures that the computationally bounded coalition in phase 1 can at most substitute its inputs by x'_C and, after obtaining its d shares, possibly stop the execution. If the coalition aborts, it possesses d shares which, by the sharing scheme's perfect privacy, contain no new information compared to the beginning. If this strategy would improve utility, aborting would be better than playing the canonical strategy in the ideal function evaluation game contradicting d-NCC. An improvement from substituting inputs and then running the protocol honestly, i. e. making all parties learn $f(x'_C, x_{-C})$, similarly contradicts d-NCC. Further, if the coalition does not substitute its inputs, then Assumption (V3) assures that there is no better strategy than running the provided secret reconstruction mechanism. Assumption (V4) is a necessary assumption in computational games which ensures that the negligible success probability of breaking a computational assumption does not affect the expected utility noticeably. It *seems* like all relevant cases have been covered. However, it is possible that the coalition substitutes its shares and then runs a different strategy than the prescribed one. Because Assumption (V3) only covers the particular setting where the coalition not substituted inputs we have no further guarantees on M's behavior. Yet, due to d-NCC and the sharing's privacy we can, essentially, deduce that it has an incentive to learn the secret and make at least one other party not learn it. This means the setting is d-NCR and we know that a utility independent mechanism (Assumption (V5)) is d-resilient in such a scenario.

Resulting from these considerations it is reasonable to ask whether we can drop the assumption for computationally bounded players. To do this, we first need to use a perfectly d-secure MPC protocol in stage 1. Such a protocol ensures that no size-d coalition, independently of its computational power, can do anything beyond substituting its inputs or prematurely aborting. Next, using a secret reconstruction mechanism which is d-resilient and utility independent in the presence of unbounded parties, leads to the same conclusion as above. Particularly, no coalition has an incentive to deviate in either step 1 or step 2, i. e. the protocol is d-resilient even assuming computationally unbounded rational coalitions. The polynomial bound on utilities can be dropped as there exists neither an assumption breakable with negligible probability nor a security parameter in which the utility could be bounded at all.

Corollary 4.3. *The strategy profile $M^* = (M_1^*, \ldots, M_n^*)$ from Construction 4.1 is a d-resilient mechanism for function evaluation in $\Gamma_{f,\mathcal{D},u}$ (Definition 3.10), if following conditions hold:*

(V1') *Function* f *is d-NCC (Definition 3.13) with respect to distribution $\mathcal{D}$ and utilities u.*

(V2') *MPC protocol* Π *computes Functionality 4.1* perfectly *d-secure (Definition 2.19).*

(V3') *The strategy profile* $M = (M_1, \ldots, M_n)$ *is a d-resilient secret reconstruction mechanism (Definition 3.5) for* $\Gamma_{S,\mathcal{D}',u}$ *with secret and auxiliary input distribution* $\mathcal{D}'$ *such that* $\Pr[\mathcal{D}' = (y, x_{[n]})] = \Pr[\mathcal{D} = x_{[n]}]$ *for all* $x_{[n]} \in X_{[n]}$ *and* $y \coloneqq f(x_{[n]})$.

(V4') *The reconstruction mechanism M is utility independent (Definition 3.8).*

Note, within the standard game theoretic notions without further computational assumptions, the notion of practical mechanisms, i. e. where the provided d-resilient equilibrium survives IDoWDS, is relevant. We claim that the function evaluation mechanism inherits the property of surviving IDoWDS from any practical d-resilient reconstruction mechanism.

Claim 4.4. *If Assumptions (V1') to (V4') hold and the reconstruction mechanism M is additionally practical, i. e. survives IDoWDS, then* $M^* = (M_1^*, \ldots, M_n^*)$ *from Construction 4.1 is a* practical *d-resilient mechanism for function evaluation in* $\Gamma_{f,\mathcal{D},u}$ *(Definition 3.10).*

Due to a lack of time, we were not able to prove Claim 4.4 formally. However, the reasonable way on how to approach this seems as follows: First, any substitution which leads to a setting which is not d-NCR, i. e. where the coalition has no incentive of learning the secret exclusively, is weakly dominated by running the honest strategy. Otherwise and equivalently, the coalition could have aborted the protocol without any interaction. Due to d-NCC this is not better than all parties learning the correct value. Hence, the only possible deviations in phase 1 left to be considered are substitutions which lead to a d-NCR setting. However, because we assume a practical reconstruction mechanism M, its strategies survive IDoWDS in any d-NCR setting. Hence, if there exists a strategy which weakly dominates M in the d-NCR setting, this contradicts the assumption of a practical mechanism. Therefore, deviating in the first phase never weakly dominates sticking to the coalition's prescription. Hence, the first step survives always IDoWDS and the second step, by assumption, does either.

Note, we have not yet stated under which circumstances the assumptions can be fulfilled, i. e. when a protocol Π and reconstruction mechanism M does exist. In the context of computationally bounded players this mainly depends on underlying communication channels and utility properties. However, we defer these considerations into Chapter 5 and focus on the formal proof of Theorem 4.2.

4.2 Proof of Theorem 4.2

In order to prove Theorem 4.2, we need to show $M^* = (M_1^*, \ldots, M_n^*)$ from Construction 4.1 is a d-resilient computational mechanism for function evaluation (Definition 3.10) given Assumptions (V1) to (V4). Particularly, this requires to prove following properties which we do in the referenced lemmas:

1. The strategy profile M^* makes all players output the *correct result* for any given types $x_{[n]} \in \mathrm{supp}(\mathcal{D})$ *within expected polynomial runtime* (Lemma 4.5).

2. M^* forms a d-resilient computational equilibrium in $\hat{\Gamma}_{f,\mathcal{D},u}$ (Lemma 4.6).

In order to simplify notation in the following, for any given strategy M_i let $M1_i$ and $M2_i$ denote the corresponding machines used in step 1 and step 2, respectively. Further, recall P_i's utility for any given strategy profile $(M_1, \ldots, M_n)$ security parameter κ is defined as expectation value over types chosen according to $\mathcal{D}(\kappa)$ and the machines' final outputs. Moreover, in the *computational*

function evaluation game the types t_i besides $x_i \in X_i$ (may) contain additional information $a_i \in \{0,1\}^*$, e. g. expressing some generated computationally hard problems. Formally we have

$$\begin{aligned} u_i(\kappa, M'_{[n]}) &\coloneqq \mathbb{E}[(x,a)_{[n]} = t_{[n]} \leftarrow \mathcal{D}(\kappa), y_{[n]} \leftarrow \mathsf{Out}^{M_{[n]}}((x,a)_{[n]}) : u_i(\kappa, f(x_{[n]}), y_{[n]})] \\ &= \mathbb{E}[(x,a)_{[n]} = t_{[n]} \leftarrow \mathcal{D}(\kappa), y_{[n]} \leftarrow \mathsf{Out}^{M2_{[n]}}(\mathsf{Out}^{M1_{[n]}}((x,a)_{[n]}) : u_i(\kappa, f(x_{[n]}), y_{[n]}))]. \end{aligned}$$

For the sake of readability we drop the additional $a_{[n]}$ and will write $x_{[n]} \leftarrow \mathcal{D}$ only. Moreover, we generally drop security parameter κ and assume every x_i and the evaluated value $f(x_{[n]})$ contains κ implicitly via its length. For sufficiently large κ we can ensure the length κ for x_i and $f(x_{[n]})$ using padding. Hence, P_i's utility from coalition C on a given x_C becomes

$$u_i(x_C, M_{[n]}) = u_i(\kappa, x_C, M_{[n]}) = \mathbb{E}[x_{[n]} \leftarrow \mathcal{D}^{|x_C}, y_{[n]} \leftarrow \mathsf{Out}^{M_{[n]}}(x_{[n]}) : u_i(f(x_{[n]}), y_{[n]})].$$

Lemma 4.5 (Correctness)**.** *If Assumptions (V2) and (V3) hold, Construction 4.1 is correct, i. e. there exists a negligible function μ and a polynomial p, such that for all types $x_{[n]} \in \mathrm{supp}(\mathcal{D})$ and all sufficiently large $\kappa \in \mathbb{N}$ we have*

1. $\Pr[\mathsf{Out}^{M^*}(x_{[n]}) = (f(x_{[n]}))^n] \geq 1 - \mu(\kappa)$*, i. e. with all but negligible probability all parties output the correct value.*
2. $\mathsf{Time}^{M^*}(x_{[n]}) \leq p(\kappa)$*, i. e. the expected runtime, in terms of interaction rounds, is polynomially bounded.*

Proof. Correct outputs in all but a negligible fraction of cases follows from the properties of Π and M: In particular, Π computes Functionality 4.1 which, by Definition 2.17, means that the parties' eventual outputs are distributed identically to the functionality's definition. Thus, for the inputs from $\mathrm{supp}(\mathcal{D}) \subseteq X_{[n]}$, we have $\mathsf{Out}^{M1^*}(x_{[n]}) \overset{\text{id.}}{\equiv} \mathsf{Share}(1^\kappa, f(x_{[n]}))$ for step 1. In the second step we use the d-resilient secret reconstruction mechanism M from Assumption (V3) to reconstruct the shared value $f(x_{[n]})$. One of M's properties is that it leads the parties with all but negligible probability to the correct secret (c. f. Definition 3.5). Hence, as desired, there exists a negligible function μ, such that

$$\Pr[\mathsf{Out}^{M^*}(x_{[n]}) = (f(x_{[n]}))^n] = \Pr[Out^M(\mathsf{Share}(1^\kappa, f(x_{[n]}))) = (f(x_{[n]}))^n] \geq 1 - \mu(\kappa),$$

where $\kappa = |x_1|$.

Expected polynomial runtime follows because Π computing a functionality requires expected polynomial runtime for any input (Definition 2.17). Additionally, the players' local outputs after step 1 are shares of the secret $f(x_{[n]})$. Given correctly computed shares, as provided by step 1, one property of a secret reconstruction mechanism is to run within expected polynomial time (c. f. Definition 3.5). Hence, combining both parts results in a total expected runtime which is polynomially bounded as well. □

Lemma 4.6 (Computational d-Resilience)**.** *If Assumptions (V1) to (V5) hold, M^* is a computational d-resilient typed equilibrium (Definition 3.2) for $\Gamma_{f,\mathcal{D},u}$.*

Proof. First, recall that a computational d-resilient equilibrium (Definition 3.2) means that no coalition C, $|C| \leq d$, gains non-negligibly from playing a different ppt strategy than M^*_C. Therefore, to prove that M^* is a computational d-resilient equilibrium, we show that for any $C \subseteq [n]$ with $|C| \leq d$, $i \in C$, and ppt ITM M'_C we have

$$u_i(\kappa, M'_C, M^*_{-C}) \lesssim u_i(\kappa, M^*), \tag{4.2}$$

where $\lesssim$ is our short notation for "there exists a negligible μ function such that for all sufficiently large κ the left expression is less or equal than the right expression plus $\mu(\kappa)$". However, it suffices to show that for all $x_C \in X_C$ we have

$$u_i(x_C, M'_C, M^*_{-C}) \lesssim u_i(x_C, M^*), \tag{4.3}$$

where $u_i(x_C, M_{[n]}) = \mathbb{E}[x_{[n]} \leftarrow \mathcal{D}^{|x_C}, y_{[n]} \leftarrow \mathsf{Out}^{M_{[n]}}(x_{[n]}) : u_i(\kappa, f(x_{[n]}), y_{[n]})]$ for any strategy profile $M_{[n]}$, i. e. P_i's expected utility for some strategy profile $M_{[n]}$ conditioned on (part of) her type being x_i. If Inequality 4.3 holds for all $x_C \in X_C$, then Inequality 4.2 holds because of

$$\begin{aligned} u_i(\kappa, M'_C, M_{-C}) &= \sum_{x_C \in X_C} \Pr[\mathcal{D}(\kappa) = (x_c, \cdot)] \cdot u_i(x_C, M'_C, M^*_{-C}) \\ &\lesssim \sum_{x_C \in X_C} \Pr[\mathcal{D}(\kappa) = (x_c, \cdot)] \cdot u_i(x_C, M^*) \\ &= u_i(\kappa, M^*). \end{aligned}$$

This holds, due to the finite and constant set X_C: The total improvement from using M'_C instead of M_C is at most $|X_C| \cdot \mu$ for some negligible μ which is a negligible function again.

Therefore, we fix an arbitrary $x_C \in X_C$ and prove Inequality 4.3. Remember, we consider strategies M'_i as split into two machines $M1'_i$ and $M2'_i$ representing step 1 and, respectively, step 2 of Construction 4.1. Further we have $M1^*_{-C} = \Pi_{-C}$ because the non-C players are supposed to stick to their strategy for computing Functionality 4.1 named g. From now on we use $M' \coloneqq (M'_C, M^*_{-C})$, $M1' \coloneqq (M1'_C, \Pi_{-C})$ and $M2' \coloneqq (M2'_C, M2^*_{-C})$ and may write

$$\begin{aligned} u_i(x_C, M') &\coloneqq \mathbb{E}[x_{[n]} \leftarrow \mathcal{D}^{|x_C}, y_{[n]} \leftarrow \mathsf{Out}^{M'}(x_{[n]}) : u_i(f(x_{[n]}), y_{[n]})] \\ &= \mathbb{E}[x_{[n]} \leftarrow \mathcal{D}^{|x_C}, y_{[n]} \leftarrow \mathsf{Out}^{M2'}(\mathsf{Out}^{M1'}(x_{[n]})) : u_i(f(x_{[n]}), y_{[n]})]. \end{aligned}$$

Since the protocol is d-secure (Assumption (V2)) and $|C| \leq d$, there exists a ppt B for $M1'_C$ such that for all $x_{[n]} \in X_{[n]}$ we have $\mathsf{Out}^{M1'}(x_{[n]}) \overset{\text{comp.}}{\equiv} \mathsf{Ideal}_{g,C,B(z)}(x_{[n]})$ with $\mathsf{Ideal}_{g,C,B(z)}(x_{[n]})$ as in Definition 2.18. It follows that $\mathsf{Out}^{M1'}(\mathcal{D}^{|x_C}) \overset{\text{comp.}}{\equiv} \mathsf{Ideal}_{g,C,B(z)}(\mathcal{D}^{|x_C})$. Before considering the particular cases of $\mathsf{Ideal}_{g,C,B(z)}(\mathcal{D}^{|x_C})$, we argue that replacing the output distribution of $M1'$ by the better analyzable ideal distribution changes the expected utility at most negligibly. Formally,

$$\begin{aligned} u_i(x_C, M') &= \mathbb{E}[x_{[n]} \leftarrow \mathcal{D}^{|x_C}, y_{[n]} \leftarrow \mathsf{Out}^{M2'}(\mathsf{Out}^{M1'}(x_{[n]})) : u_i(f(x_{[n]}), y_{[n]})] \\ &\approx \mathbb{E}[x_{[n]} \leftarrow \mathcal{D}^{|x_C}, y_{[n]} \leftarrow \mathsf{Out}^{M2'}(\mathsf{Ideal}_{g,C,B(z)}(x_{[n]})) : u_i(f(x_{[n]}), y_{[n]})]. \end{aligned} \tag{4.4}$$

where $\approx$ is our short notation for the compared terms differing at most negligibly. Given relation 4.4, upper-bounding the last term by $u_i(x_C, M^*) + \mu(\kappa)$ for some negligible μ finishes the proof. It holds due to two reasons. First, as long as the interaction of protocol $M2'$ runs in expected polynomial time, we have

$$\mathsf{Out}^{M2'}(\mathsf{Out}^{M1'}(\mathcal{D}^{|x_C})) \overset{\text{comp.}}{\equiv} \mathsf{Out}^{M2'}(\mathsf{Ideal}_{g,C,B(z)}(\mathcal{D}^{|x_C})).$$

Otherwise a ppt D could distinguish the inner random variables by simulating the interaction of $M2'$ sufficiently long on a given challenge. More formally, a ppt can choose a polynomial p such that with all but negligible probability the expected polynomial time interaction stops within $p(\kappa)$ steps. Second, a property of MPC utilities (Assumption (V4)) is, that they are polynomially bounded in security parameter κ within the computational setting. Therefore, the expected utility changes at most negligibly on computationally indistinguishable distributions. Indeed the

expectation value of any polynomially bounded function on computationally indistinguishable inputs changes at most negligibly. We formalize this in Lemma 2.2. The argument is that a non-negligibly higher expectation value implies the existence of an element, hit with non-negligible probability, on which the random variables differ non-negligibly. The distinguisher then takes advantage of this element.

By Definition 2.18, $\mathsf{Ideal}_{g,C,B(z)}(\mathcal{D}^{|x_C})$ splits into two scenarios where either prematurely aborting is allowed to a coalition or is not. We suppose it is allowed as this covers the other scenario as special case. Depending on B's random choices, $\mathsf{Ideal}_{g,C,B(z)}(\mathcal{D}^{|x_C})$ is distributed identical to one of the following cases:

1. $((\perp)^{|-C|}, B(x_C, x'_C, g(x'_C, \mathcal{D}^{|x_C}_{-C})_C, \perp))$ if B aborts the protocol after choosing some value $x'_C \in X_C$ and obtaining its outputs. We assume that B never aborts after inserting an element $x'_C \notin X_C$ as this case is covered by case 2a. Plugging in functionality g's definition, we get $((\perp)^{|-C|}, B(x_C, x'_C, s_C, \perp))$ with $s_C \leftarrow \mathsf{Share}(f(x'_C, \mathcal{D}^{|x_C}_{-C}))_C$.

2. $(g(x'_C, \mathcal{D}^{|x_C}_{-C})_{-C}, B(x_C, x'_C, g(x'_C, \mathcal{D}^{|x_C}_{-C})_C))$ for $B(x_C) = x'_C$ if B not aborts the protocol after obtaining its outputs. By definition of functionality g (4.1) we subdivide this into the cases

   (a) $((\perp)^{|-C|}, B(x_C, x'_C, (\perp)^{|C|}))$ if $x'_C \notin X_C$.

   (b) $(s_{-C}, B(x_C, x_C, s_C))$ with $(s_C, s_{-C}) \leftarrow \mathsf{Share}(f(x_C, \mathcal{D}^{|x_C}_{-C}))$ if $x'_C = x_C$.

   (c) $(s_{-C}, B(x_C, x'_C, s_C))$ with $(s_C, s_{-C}) \leftarrow \mathsf{Share}(f(x'_C, \mathcal{D}^{|x_C}_{-C}))$ if $x_C \neq x'_C \in X_C$.

Let E_1 denote the event when B's actions lead to case 1. Analogously, E_2 denotes the event for case 2a, E_3 for case 2b, and E_4 for case 2c, respectively. Further, let $\mathsf{Ideal}^{|E_j}_{g,C,B(z)}(\mathcal{D}^{|x_C})$ denote the ideal distribution conditioned on event E_j. Using conditioned probabilities we get

$$\begin{aligned}&\mathbb{E}[x_{[n]} \leftarrow \mathcal{D}^{|x_C}, y_{[n]} \leftarrow \mathsf{Out}^{M2'}(\mathsf{Ideal}_{g,C,B(z)}(x_{[n]})) : u_i(f(x_{[n]}), y_{[n]})] \\ &\quad = \sum_{j\in[4]} \Pr[E_j] \cdot \mathbb{E}[x_{[n]} \leftarrow \mathcal{D}^{|x_C}, y_{[n]} \leftarrow \mathsf{Out}^{M2'}(\mathsf{Ideal}^{|E_j}_{g,C,B(z)}(x_{[n]})) : u_i(f(x_{[n]}), y_{[n]})],\end{aligned} \tag{4.5}$$

for $\sum_{j\in[4]} \Pr[E_j] = 1$. We prove that each expectation value from the sum above is bounded by $u_i(x_C, M^*) + \mu_j(\kappa)$ for some negligible μ_j.

Event E_1: In this case B aborted after inserting some value $x'_C \in X_C$ and obtaining its shares $s_C \leftarrow \mathsf{Share}(f(x'_C, \mathcal{D}^{|x_C}_{-C}))_C$ while every honest player obtains $\perp$. We construct a strategy for the ideal evaluation game which results in an output distribution over Y^n which is computationally indistinguishable from the real execution conditioned on E_1. Hence, this strategy's utility deviates at most negligibly from the utility gained in the real game under event E_1. Due to the d-NCC assumption (Assumption (V1)) this strategy is upper bounded and, thus, the real utility is upper bounded, too.

Recall the ideal function evaluation setting from Definition 3.12: First, $x_{[n]} \leftarrow \mathcal{D}$ is sampled and every player P_i obtains x_i. Then, every P_i samples $x'_i \leftarrow \delta^1_i(x_i) \in \Delta(X_i \cup \{\perp\})$ and sends x'_i to the mediator. If no $\perp$ was input the mediator returns $f(x'_{[n]})$ to every player. Otherwise, every P_i is recommended the likeliest correct function value with respect to type distribution $\mathcal{D}^{|x'_i}$. Given the recommended value, every player samples and outputs a value $y_i \leftarrow \delta^2_i(x_i, x'_i, y_i)$. The d-NCC assumption ensures that the canonical strategy $\delta^*_{[n]}$, i. e. every player truthfully providing x_i to the mediator and outputting the recommendation, is a d-resilient equilibrium.

We construct an ideal game strategy δ_C for coalition C which together with δ^*_{-C} leads to an output distribution over Y^n computationally indistinguishable from the real game conditioned on E_1. Let $\delta^1_C(x_C) \coloneqq \bot$. This ensures that every non-C player P_i is recommended the likeliest function value with respect to $\mathcal{D}^{|x_i}$ exactly as computed by $M2^*_i$ in case of event E_1, i. e. when obtaining $\bot$ from $M1^*_i$ (c. f. Definition 3.12 and Construction 4.1). Therefore, it remains to ensure that C's outputs from the ideal strategy are computationally indistinguishable from the real one. Therefore, $\delta^2_C(x_C, x'_C, \bot)$ simulates and outputs $\mathsf{Out}^{M2'}((\bot)^{|-C|}, B(t_C, \mathsf{Share}(y)_C, \bot))_C$ for an arbitrary $y \in Y$. B's inputs are distributed as in the real execution because for any non-qualified set like C we have $\mathsf{Share}(f(x'_C, x_{-C}))_C \overset{\text{id.}}{\equiv} \mathsf{Share}(y)_C$ by the secret-sharing scheme's perfect privacy. However, because the interaction of $M2'$ has no runtime limit δ^2_C needs to stop its simulation after a certain number of steps. This point can be chosen such that with all but negligible probability the interaction finishes. If the simulation not ends in time δ_2 outputs an arbitrary element $y \in Y$. Altogether, we get

$$\mathsf{Out}^{\delta_C, \delta^*_{-C}}(\mathcal{D}^{|x_C}) \overset{\text{comp.}}{\equiv} \mathsf{Out}^{M2'}(\mathsf{Ideal}^{|E_1}_{g,C,B(z)}(\mathcal{D}^{|x_C}))$$

where we leave $(\delta_C, \delta^*_{-C})$'s interaction with the mediator implicit. Thus,

$$\begin{aligned}
&\mathbb{E}[x_{[n]} \leftarrow \mathcal{D}^{|x_C}, y_{[n]} \leftarrow \mathsf{Out}^{M2'}(\mathsf{Ideal}^{|E_1}_{g,C,B(z)}(x_{[n]})) : u_i(f(x_{[n]}), y_{[n]})] \\
&\qquad \lesssim \mathbb{E}[x_{[n]} \leftarrow \mathcal{D}^{|x_C}, y_{[n]} \leftarrow \mathsf{Out}^{\delta_C, \delta^*_{-C}}(x_{[n]}) : u_i(f(x_{[n]}), y_{[n]})] \\
&\qquad = u_i(x_C, \delta_C, \delta^*_{-C}) \\
&\qquad \lesssim u_i(x_C, \delta^*_{[n]}) \\
&\qquad \lesssim u_i(x_C, M^*)
\end{aligned}$$

as desired. By Lemma 2.2, swapping computationally indistinguishable distributions used as input for a polynomially bounded function as u_i (Assumption (V4)) changes the corresponding expected value negligibly. Hence, we get the first inequality and rewrite the expectation value by $u_i(x_C, \delta_C, \delta^*_{-C})$. Because of d-NCC (Assumption (V1)) we have that $\delta^*_{[n]}$ is a d-resilient equilibrium which yields the second-last inequality. Strategy $\delta^*_{[n]}$ always makes every party output the correct result whereas M^* by correctness (Lemma 4.5) guarantees the same only with all but negligible probability lemma:constr-correct. Again, this negligible difference changes the expected utility only negligibly because u_i is polynomially bounded (Lemma 2.2), leading to the last inequality.

Event E_2: The case of event E_2 where B does not abort but inserts $\bot$ and, thus, obtains $\bot$ as return value, is handled analogue to the previous case. The only difference is that $\delta^2_C(x_C, \bot, \bot)$ runs and outputs $\mathsf{Out}^{M2'}((\bot)^{|-C|}, B(x_C, x'_C, (\bot)^{|C|}))_C$ instead of using B on some generated shares.

Event E_3: In case of event E_3, B did not abort and sticked to her true value x_C. Therefore, the parties' inputs for step 2 are shares distributed according to $\mathsf{Share}(f(\mathcal{D}^{|x_C}))$. By Assumption (V3) strategy profile (M_C, M_{-C}) constitutes a computational d-resilient practical secret reconstruction mechanism (Definition 3.5) for $\hat{\Gamma}_{S,\mathcal{D}',u}$ where distribution $\mathcal{D}'$ outputs secret y and auxiliary information $x_{[n]}$ according to $x_{[n]} \leftarrow \mathcal{D}$ and $y \coloneqq f(x_{[n]})$. Then shares $s_{[n]} \leftarrow \mathsf{Share}(f(y))$ are sampled and every player P_i obtains type (s_i, x_i). Note, the players' inputs to $(M2'_C, M2^*_{-C})$ from the ideal distribution conditioned on E_3 are distributed exactly as in this reconstruction game. Furthermore, because $M1_{-C}$'s outputs are not $\bot$, $M2^*_{-C}$ runs the equilibrium strategy

M_{-C} (c. f. Construction 4.1). Thus, we get

$$\begin{aligned}
&\mathbb{E}[x_{[n]} \leftarrow \mathcal{D}^{|x_C}, y_{[n]} \leftarrow \mathsf{Out}^{M2'_C, M_{-C}}(\mathsf{Ideal}^{|E_3}_{g,C,B(z)}(x_{[n]})) : u_i(f(x_{[n]}), y_{[n]})] \\
&\quad = \mathbb{E}[x_{[n]} \leftarrow \mathcal{D}^{|x_C}, s_{[n]} \leftarrow \mathsf{Share}(f(x_{[n]})), y_{[n]} \leftarrow \mathsf{Out}^{M2'_C, M_{-C}}((s,x)_{[n]}) : u_i(f(x_{[n]}), y_{[n]})] \\
&\quad = u_i(x_C, M2'_C, M_{-C}) \\
&\quad \lesssim u_i(x_C, M_C, M_{-C}) \\
&\quad \leq (1-\mu(\kappa)) \cdot \mathbb{E}[x_{[n]} \leftarrow \mathcal{D}^{|x_C}, y \coloneqq f(x_{[n]}) : u_i(y, y^n)] + \mu(\kappa)(\max_{y, y_{[n]} \in Y \times Y^n} u_i(y, y_{[n]})) \\
&\quad \approx \mathbb{E}[x_{[n]} \leftarrow \mathcal{D}^{|x_C} : u_i(f(x_{[n]}), (f(x_{[n]}))^n)] \\
&\quad \approx \mathbb{E}[x_{[n]} \leftarrow \mathcal{D}^{|x_C}, y_{[n]} \leftarrow \mathsf{Out}^{M^*}(x_{[n]}) : u_i(f(x_{[n]}), (f(x_{[n]}))^n)] \\
&\quad = u_i(x_C, M^*).
\end{aligned}$$

The first equality follows by plugging in the ideal distribution's behavior conditioned on E_3, i. e. the parties obtain a correctly sampled secret-sharing of the true function value. To simplify notation we dropped the application of B on C's inputs to the second phase, which is reasoned by assuming that $M2'_C$ applies B on its inputs. This expected value equals P_i's utility when obtaining auxiliary information x_C within the secret reconstruction game $\hat{\Gamma}_{S,\mathcal{D}',u}$ which is denoted by $u_i(x_C, M2'_C, M_{-C})$. Because (M_C, M_{-C}) is a d-resilient equilibrium in $\hat{\Gamma}_{S,\mathcal{D}',u}$ (Assumption (V3)) the next inequality follows. More precisely, Assumption (V3)) states that (M_C, M_{-C}) is a reconstruction mechanism. Thus, for any shared secret $y \leftarrow f(\mathcal{D}^{|x_C})$ every player outputs the correct value with all but negligible probability μ. P_i then gains with probability μ an arbitrary utility but at most $\max_{y,y_{[n]}} u_i(y, y_{[n]})$. Because the utilities are polynomially bounded (Assumption (V4)) the product with μ is negligible again and we estimate the sum with $\mathbb{E}[x_{[n]} \leftarrow \mathcal{D}^{|x_C} : u_i(f(x_{[n]}), (f(x_{[n]}))^n)]$. Running M^* leads to the correct outputs with all but negligible probability by its correctness property (Lemma 4.5). This negligible probability changes the expected utility at most negligibly because the utilities are polynomially bounded, i. e. Lemma 2.2 applies again. This leads to the last expected value which equals $u_i(x_C, M^*)$ as desired.

Event E_4: In case of E_4, B changed her value to some $x'_C \in X_C$ and did not abort. Thus, the parties obtain shares distributed according to $\mathsf{Share}(f(x'_C, \mathcal{D}^{|t_C}_{-C}))$ from step 1 where $\mathcal{D}^{|x_C}_{-C}$ is the distribution on $\mathcal{D}^{|x_C}$ restricted to $-C$'s outputs. We want to upper-bound the expected utility from this replacement strategy by $u_i(x_C, M^*)$. To simplify notation in the following, we use $y' \coloneqq f(x'_C, x_{-C})$, $y^* \coloneqq f(x_C, x_{-C}) = f(x_{[n]})$ and $x'_{-C} \coloneqq x_{-C}$ where x_{-C} is defined by the context. Formally, we show the following inequality

$$\begin{aligned}
&\mathbb{E}[x_{[n]} \leftarrow \mathcal{D}^{|x_C}, y_{[n]} \leftarrow \mathsf{Out}^{M2'}(\mathsf{Ideal}^{|E_4}_{g,C,B(z)}(x_{[n]})) : u_i(y^*, y_{[n]})] \\
&\quad = \mathbb{E}[x_{[n]} \leftarrow \mathcal{D}^{|x_C}, s_{[n]} \leftarrow \mathsf{Share}(y'), y_{[n]} \leftarrow \mathsf{Out}^{M2'}((x, x', s)_{[n]})) : u_i(y^*, y_{[n]})] \\
&\quad \lesssim u_i(\kappa, M^*),
\end{aligned} \tag{4.6}$$

where $x'_C \neq x_C$ is given to $M2'_C$ as explicit state information representing its additional knowledge from $M1'_C$ due to event E_4. For the sake of contradiction assume Inequality 4.6 does not hold, i. e. there exists a non-negligible function $p\colon \mathbb{N} \to \mathbb{R}$ such that

$$\begin{aligned}
&\mathbb{E}[x_{[n]} \leftarrow \mathcal{D}^{|x_C}, s_{[n]} \leftarrow \mathsf{Share}(y'), y_{[n]} \leftarrow \mathsf{Out}^{M2'}((x, x', s)_{[n]}) : u_i(y^*, y_{[n]})] \\
&\qquad \geq u_i(\kappa, M^*) + p(\kappa).
\end{aligned} \tag{4.7}$$

We show, this is impossible due to d-NCC (Assumption (V1)) as well as the secret reconstruction mechanism's d-resiliency (Assumption (V3)) and utility independence (Assumption (V5)).

As we already have seen when analyzing event E_1, guessing y^* independently of $-C$'s inputs cannot increase the expected utility by d-NCC. Additionally, d-NCC implies that C's expected utility cannot be increased by running the secret reconstruction strategy M_C to obtain y' and, afterwards, guessing y^* based on y'. This holds because running M_C makes every party in $-C$ learn and output y' exactly as in the ideal function evaluation game when x_C was replaced by x'_C. Then, by d-NCC, no strategy which C possibly runs (within $M2'_C$) to compute its final guess for y^* yields a utility gain. Indeed, in order to fulfill Inequality 4.7, $M2'_C$ takes advantage from learning y' with non-negligible probability while the non-colluding players $-C$ do not. However, given such an $M2'_C$ we can construct a (non-negligibly) better strategy than M_C for another secret reconstruction setting contradicting the mechanism's equilibrium properties (Assumption (V3)) due to its utility independence (Assumption (V5)). Therefore Inequality 4.7 may not hold and we get Inequality 4.6.

More formally, we construct an ideal function evaluation game strategy (δ_C^1, δ_C^2) as follows: First we set $\delta_C^1(x_C) = x'_C$, i. e. the coalition sends the changed value x'_C to the mediator. Given the mediator's recommendation $y' = f(x'_C, x_{-C})$, $\delta_C^2(x_C, x'_C, y')$ samples $s_{[n]} \leftarrow \mathsf{Share}(y')$, simulates the interaction of $M2' = (M2'_C, M_{-C})$ on $(x, x', s)_{[n]}$ with $x'_{-C} \coloneqq x_{-C}$ sufficiently long and returns the outputs of $M2'_C$. As previously, sufficiently long means that with all but negligible probability the simulated interaction ends. In case it does not end within the required steps, δ_C^2 outputs an arbitrary $y_C \in Y^{|C|}$. Because the utility functions are polynomially bounded this changes the expected utility at most negligibly (Lemma 2.2). Thus, using the mediated game's canonical strategy δ^* which is a d-resilient equilibrium by d-NCC, we have

$$\begin{aligned}
&u_i(x_C, \delta_C, \delta^*_{-C}) \\
&\quad \approx \mathbb{E}[x_{[n]} \leftarrow \mathcal{D}^{|x_C}, s_{[n]} \leftarrow \mathsf{Share}(y'), y_C \leftarrow \mathsf{Out}^{M2'}((x, x', s)_{[n]})_C : u_i(y^*, (y')^{|-C|}, y_C)] \\
&\quad \lesssim u_i(x_C, \delta_C^*, \delta^*_{-C}) \\
&\quad \approx u_i(x_C, M^*).
\end{aligned}$$

Together with Inequality 4.7 and the expectation value's linearity we have

$$\begin{aligned}
&\mathbb{E}[x_{[n]} \leftarrow \mathcal{D}^{|x_C}, s_{[n]} \leftarrow \mathsf{Share}(y'), y_{[n]} \leftarrow \mathsf{Out}^{M2'}((x, x', s)_{[n]}) : u_i(y^*, y_C, y_{-C})] \\
&\quad - \mathbb{E}[x_{[n]} \leftarrow \mathcal{D}^{|x_C}, s_{[n]} \leftarrow \mathsf{Share}(y'), y_C \leftarrow \mathsf{Out}^{M2'}((x, x', s)_{[n]})_C : u_i(y^*, (y')^{|-C|}, y_C)] \\
&\quad = \mathbb{E}[x_{[n]} \leftarrow \mathcal{D}^{|x_C}, s_{[n]} \leftarrow \mathsf{Share}(y'), y_C \leftarrow \mathsf{Out}^{M2'}((x, x', s)_{[n]}) : u_i(y^*, y_C, y_{-C}) - u_i(y^*, (y')^{|-C|}, y_C)] \\
&\quad \gtrsim u_i(\kappa, M^*) + p(\kappa) - u_i(\kappa, M^*) \\
&\quad = p(\kappa).
\end{aligned}$$

Because this difference is non-negligible it follows that with non-negligible probability some parties in $-C$ not output the shared value y' in the *real* game. However, as we have seen when analyzing event E_1, to gain utility and, hence, fulfill Inequality 4.7, the output y_C of $M2'_C$ needs to depend on y' in at least some of these cases. Hence, C has the incentive to learn the shared value y' while preventing the other parties from doing so. Especially, C has the incentive to reconstruct secret y' in a reconstruction game where, essentially, secrets are chosen according to $f(x'_C, \mathcal{D}^{|x_C}_{-C})$. Because we assume the reconstruction mechanism M from Assumption (V3) to be utility independent (Assumption (V5)), M is a d-resilient equilibrium within any setting which is d-non-cooperatively secret reconstructable. However, if Inequality 4.7 holds, then $M2'_C$ is a non-negligibly better alternative compared to M_C in this setting. Particularly, the utility gain is polynomially related to the non-negligible $p(\kappa)$. This contradicts Assumption (V5) and, hence, Inequality 4.7 does not hold. Therefore we get Inequality 4.6 as desired. □

5 Instantiating Construction 4.1 — Possibilities and Impossibilities

In Chapter 4 we provide the modularized Construction 4.1 which forms a (computational) d-resilient function mechanism under certain assumptions (c. f. Theorem 4.2 and Corollary 4.3). Particularly, we made the assumption that the given setting is d-NCC (Definition 3.13) which is necessary for the existence of a d-resilient function evaluation mechanism. In the computational case we, additionally, assumed that the utilities are polynomially bounded which is also inevitable when relying on computationally hard problems. Otherwise, trying to solve the underlying computational problems, typically, would improve the utility compared to running the desired protocol honestly. Beyond these *necessary* assumptions, we made assumptions regarding Construction 4.1's components. For the computational d-resilient mechanism we, for example, require that the MPC protocol d-securely computes Functionality 4.1 in the first phase. Additionally, the protocol used within the second phase is required to be a utility independent computational d-resilient reconstruction mechanism. Yet, we have not stated under which circumstances such protocols with the required properties do exist, i. e. we have not given any conditions for the existence of a function evaluation mechanism.

Whether the components can be instantiated such that they match our assumptions depends on the given setting. Concretely, beyond the necessary conditions on input distribution and utilities, the possibility or impossibility of instantiating Construction 4.1 as (possibly computational) function evaluation mechanism depends on:

- The allowed size d of coalitions in relation to the number n of active players.
- Using private or public communication channels.
- Using simultaneous or non-simultaneous communication channels.

5.1 Existence of Secure MPC Protocols

We first cover the existence of d-secure MPC protocols. Since the seminal works of Yao [Yao86] and Goldreich, Micali, and Wigderson [GMW87], the field of traditionally secure MPC has attracted the attention of many researchers. Over the years, the existence of secure MPC protocols has been analyzed and characterized with respect to various assumptions and communication models. Dodis and Rabin [DR07, Theorem 1.2] provide a good overview over existing results. In following theorem we state two results which are important for our setting.

Theorem 5.1. *Let f be an n-ary functionality.*

1. *A d-secure protocol computing f exists for any $d < n$, assuming a broadcast channel, private point-to-point channels, and a certain cryptographic assumption (e. g. enhanced trapdoor permutations within the construction of Goldreich [Gol04, Chapter 7.5]).*

2. *A* perfectly *d-secure protocol computing f exists for any $d < \frac{n}{3}$, assuming private point-to-point channels ([BGW88]).*

Note, the assumptions are idealized, i. e. perfect broadcast or private communication channels, generally, cannot be expected in reality but need to be instantiated using cryptographic means. However, making these assumptions helps separating issues arising in communication channels from designing the protocols itself. In the traditional setting, in most cases, ideal channels are securely replaced by implementations which behave non-noticeably different from the ideal one for computationally bounded parties. For instance, private channels are replaced by public channels and an existing public key infrastructure used for encryption and signatures. As noted when discussing IDoWDS within computational games, with respect to some game-theoretic notions this is not as straightforward (c. f. Section 3.1).

5.2 Existence of Utility Independent Reconstruction Mechanisms

For rational secret-sharing and reconstruction the results are neither as standardized nor as numerous as in traditional MPC. Therefore, additionally to just collecting results, it is necessary to check whether results apply to our setting. The strong assumption of utility independence (Definition 3.8) has been analyzed comprehensively by Asharov and Lindell [AL11] and, based on their results, by De and Pal [DP13]. In their model they take into account players who have, independently of their own outputs, an incentive to make other players output wrong values. This is reflected by allowing the parties to output a value $\perp$ instead of a guess on the secret. This can be seen as an attempt to model the players' confidence on their outputs within the utility functions: If P_i's chance of outputting a wrong value is too high with respect to the current knowledge, P_i rather outputs $\perp$ than some guess. Because their model is more general than ours, their possibility results hold in our setting as well. They provide a protocol which is a d-resilient reconstruction mechanism as long as the coalitions are in a strict minority, i. e. $d < \frac{n}{2}$. The special case of $n = 2$ is excluded because this leads to the trivial statement of 0-resilience. We restate Theorem 5.2 of Asharov and Lindell [AL11] using our notions.

Theorem 5.2. *Let $d, n \in \mathbb{N}$ with $n \geq 3$ and $n \geq d > 2$. A utility independent $(\lceil \frac{d}{2} \rceil - 1)$-resilient practical reconstruction mechanism exists for any perfectly $(\lceil \frac{d}{2} \rceil - 1)$-authenticated $(d-1, n)$-secret-sharing scheme, assuming a* simultaneous *broadcast channel and private point-to-point channels.*

In contrast to Asharov and Lindell [AL11, Theorem 5.2], we explicitly state their assumption of having perfectly authenticated shares which is often left implicit in rational secret-sharing. This is relevant when instantiating the schemes and making formal statements on these. In particular, perfect d-authentication with ppt reconstruction is, to the best of our knowledge, only possible for $d < \frac{n}{3}$ and can, for example, be realized using Shamir's secret-sharing scheme (c. f. [CDN15]). For $\frac{n}{3} < d < \frac{n}{2}$ there are results regarding the robustness of secret-sharing schemes which detect wrong shares with very high probability [RB89]. However, in the presence of unbounded parties, this precision still might be too weak. Yet, for *any* $d < n$ we can instantiate *computational* d-authenticated schemes using (d, n)-secret-sharing schemes and secure digital signatures (c. f. Definition 2.16). For computational d-authenticated schemes, the construction

of Asharov and Lindell [AL11] preserves its properties as *computational* mechanism. Hence, if we consider the overlaps from the existence of perfectly (resp., computationally) d-authenticated secret-sharing schemes with the maximal boundary of $d < \frac{n}{2}$ from Theorem 5.2, we get following Corollary 5.3.

Corollary 5.3. *Let $n \in \mathbb{N}$ with $n \geq 3$. Assuming a simultaneous broadcast channel and private point-to-point channels, then*

1. *A* perfectly $(\lceil \frac{n}{3} \rceil - 1)$*-authenticated* $(n-1, n)$*-secret-sharing scheme and a corresponding utility independent* $(\lceil \frac{n}{3} \rceil - 1)$*-resilient practical reconstruction mechanism exists.*

2. *A* computational $(\lceil \frac{n}{2} \rceil - 1)$*-authenticated* $(n-1, n)$*-secret-sharing scheme and a corresponding utility independent* computational $(\lceil \frac{n}{2} \rceil - 1)$*-resilient reconstruction mechanism exists, assuming, additionally, a secure digital signature scheme.*

While private channels are a rather standard assumption within MPC, simultaneous channels are not. Such channels are hard or even impossible to realize. Rational secret-sharing and reconstruction in a non-simultaneous communication model was first studied by Kol and Naor [KN08]. They propose a protocol which depends on the concrete utility values and, informally, works as long as the parties do not have too much auxiliary information on the particular shared secret. Indeed, Asharov and Lindell [AL11, Theorem 4.13] have proven the impossibility of a secret reconstruction mechanism within the non-simultaneous model for *arbitrary* auxiliary inputs even assuming "coalitions" of size 1. However, in our concrete scenario we have auxiliary information which *may* help the parties to recognize the secret. For instance, within the function evaluation setting a party might know due to her inputs, that the shared value, i. e. the evaluated value, is one of two secrets. Although their protocol is utility dependent, we reference this result here, as full utility independence is probably a too strict requirement for the mechanism's existence. In subsequent works it might be checked formally, whether in some settings with little auxiliary information on the shared secret, i. e. the function's value, the protocol of Kol and Naor [KN08] can be used and, thus, whether non-simultaneous communication in some cases suffice.

Besides the impossibility for utility independent reconstruction within the non-simultaneous model, they prove the impossibility for utility independent d-resilient reconstruction with $d \geq \frac{n}{2}$ ([AL11, Theorem 5.3]). Both impossibility results do not rely on their stronger utility model and, thus, apply to our setting.

Theorem 5.4. *Let $d, n \in \mathbb{N}$ and $d \leq n$. For* any $(d-1, n)$*-secret-sharing scheme*

1. *There exists no* utility independent, 1*-resilient computational reconstruction mechanism,* assuming non-simultaneous channels.

2. *There exists no* utility independent $(\lceil \frac{n}{2} \rceil)$*-resilient computational reconstruction mechanism.*

As their theorems show, the property of utility independent reconstruction is a strong one even assuming simultaneous channels. Assuming simultaneous channels and given the concrete utility functions, it is possible to instantiate computational d-resilient reconstruction mechanisms for any d-authenticated (d, n)-secret-sharing scheme. Among others, Abraham et al. [Abr+06] and Gordon and Katz [GK06] propose such utility dependent mechanisms. However, we were not able to distill common properties which would allow us to use them as the second subcomponent of Construction 4.1. Nevertheless, we still conjecture it is possible to find and formalize such a property in our model and leave this as open problem for the future.

5.3 Implications for Construction 4.1

Theorem 5.5, essentially, summarizes the results from the previous sections with respect to the existence of a d-resilient function evaluation mechanism, i. e. a secure instantiation of Construction 4.1. Again, d-NCC needs to be stated as an explicit requirement in the following. Otherwise, with respect to our definitions, no function evaluation mechanism can exist in the given setting.

Theorem 5.5. *Let $n \in \mathbb{N}$ with $n \geq 3$. Further let function $f\colon X_{[n]} \to Y$, distribution $\mathcal{D} \in \Delta(X_{[n]})$ and utilities $u_i\colon Y \times Y^n \to \mathbb{R}$ for $i \in [n]$.*

1. *If f is $(\lceil \frac{n}{3} \rceil - 1)$-NCC with respect to $\mathcal{D}$ and $u_{[n]}$ (Definition 3.13), then a $(\lceil \frac{n}{3} \rceil - 1)$-resilient function evaluation mechanism exists, assuming a simultaneous broadcast channel and private point-to-point channels.*

2. *If f is $(\lceil \frac{n}{2} \rceil - 1)$-NCC with respect to $\mathcal{D}$ and $u_{[n]}$ (Definition 3.13), then a* computational *$(\lceil \frac{n}{2} \rceil - 1)$-resilient function evaluation mechanism exists, assuming polynomially bounded utilities, a simultaneous broadcast channel, private point-to-point channels and secure digital signatures.*

Proof. The first part follows due to Corollary 4.3, Theorem 5.1 and Corollary 5.3. Recall, Corollary 4.3 states that a $(\lceil \frac{n}{3} \rceil - 1)$-resilient practical mechanism exists if Assumptions (V1') to (V4') hold:

1. Assumption (V1'), i. e. the setting is $(\lceil \frac{n}{3} \rceil - 1)$-NCC: Given as (necessary) condition within Theorem 5.5.

2. Assumption (V2'), i. e. a *perfectly* $(\lceil \frac{n}{3} \rceil - 1)$-secure MPC protocol exists and is used in Construction 4.1: Under the given assumptions, such a protocol exists by part (2) of Theorem 5.1 for any value less than $\frac{n}{3}$.

3. Assumptions (V3') and (V4'), i. e. a utility independent $(\lceil \frac{n}{3} \rceil - 1)$-resilient secret reconstructing mechanism exists and is used in Construction 4.1: Under the given assumptions, such a mechanism exists for any value less than $\frac{n}{3}$ exists by part (1) of Corollary 5.3.

The second part follows analogously. □

6 Conclusions

In this thesis we focused on the MPC problem of securely evaluating a function f in the presence of n rational players having private inputs and natural incentives. A major goal was to provide a formal proof regarding the existence of d-resilient function evaluation mechanisms, i. e. protocols leading every party to the correct function value while being stable against coalitions of up to size d. Indeed, we have formally proven the existence of such a mechanism under certain assumptions with respect to rational parties forming coalitions of (1) size up to $(\lceil \frac{n}{3} \rceil - 1)$ for computationally unbounded parties and (2) size up to $(\lceil \frac{n}{2} \rceil - 1)$ for computationally bounded parties. The assumptions covered necessary assumptions regarding the players' utilities and input distributions, but also the communication channels. Indeed, these results do only hold assuming, especially, a simultaneous broadcast channel.

In order to prove our existence theorem we, first, established a game-theoretic model suiting the cryptographic needs. This was a comprehensive task as most works in the field of rational cryptography leave many assumed details implicit. An additional problem was that there has been no common agreement on (de facto) standard notions, yet. However, the resulting model is general enough to model the games for secret reconstruction as well as function evaluation and might be adopted to cover different problems in the future.

Next, we constructed a generic protocol (Construction 4.1), forming a d-resilient function evaluation mechanism under clearly stated and well-defined assumptions. The decision of providing and analyzing a generic construction resulted from a theoretical as well as a practical point of view. For practical realizations it is often useful and guarantees certain flexibility to be able to exchange used components, for example, in order to gain speed ups. As long as these components satisfy our assumptions, this replacement can be done without further complications. From the theoretical viewpoint, distilling properties of existing notions which are sufficient and possibly necessary to construct a function evaluation mechanism yields insights on the considered problem itself.

Eventually, we collected existing results which made statements towards the existence of our assumptions. In particular this comprises the existence of traditionally d-secure MPC protocols and the existence of utility independent, d-resilient secret reconstruction mechanisms. While d-secure MPC protocols, under suitable computational assumptions, do exist for any $d < n$, utility independence is a rather strong notion. Particularly, due to utility independence we were only able to make statements on the existence for coalitions forming strict minorities. Yet, any utility independent reconstruction mechanism published in the future may be used within implementations of Construction 4.1.

6.1 Future Work

During writing this thesis some open problems, tasks, and questions came up which we were not able to address satisfactorily, yet. First of all, a major point is the existence of a function evaluation mechanism for settings where more than the strict minority of parties collude. This, especially, includes the important special case where two players want to evaluate a given function. In the current construction or, more precise, under the current assumption of a *utility independent* secret reconstruction mechanism, our construction cannot be instantiated for two players due to the impossibility proof of [AL11, Theorem 5.3]. We conjecture that the requirement for utility independence is too strong. Particularly, we believe, that considering Construction 4.1 with a specific secret reconstruction mechanism (e. g. [GK06]) instead of a general one can overcome this problem. If this is possible, an interesting task is to characterize minimal requirements on the secret reconstruction mechanism for instantiating Construction 4.1 as (computational) d-resilient function evaluation mechanism.

Besides the existence in other settings, another interesting direction is to check non-existence of function evaluation mechanism. In particular this is related to the impossibility theorem of Asharov and Lindell [AL11] who prove the non-existence of utility independent mechanisms given non-simultaneous channels and arbitrary auxiliary information.

Another open problem is the extension of the notion of IDoWDS to computational games. As discussed in Section 3.1, the straightforward extension is problematic. Indeed, IDoWDS is susceptible to "last" rounds where a computational problem is solved with certainty, although these rounds, practically, are never reached. Our idea, which we briefly explained in Section 3.1, is to sample for every player a certain round after which she gains no utility anymore. This round can be seen as modeling the point in time when the player's cost for running the machine exceeds the utility gained from learning a value. The rounds should be chosen such that they allow the players to finish the protocol with almost all but negligible probability within the given bound. However, the point in time should be polynomial in the security parameter, in order to guarantee the computational hardness of underlying problems. Further, if these rounds are chosen such that the players mutually cannot predict when another player stops, the players have no common round in which they all stop sending beneficial information. This seems to prevent the backwards induction argument and possibly restores a meaningful notion of IDoWDS. If the notion of IDoWDS is restored in the computational setting, then it seems possible to prove composition theorems which allow, for example, to replace private channels by public key encryption and a public key infrastructure. With respect to the straightforward definition which were often (implicitly) assumed, this composition fails. Besides IDoWDS in the computational scenario, we were not able to provide a formal proof our construction survives IDoWDS with respect to computationally unbounded players. Particularly, formally proving Claim 4.4 is an important open subsequent tasks.

Last but not least, in this thesis we mainly focused on a function evaluation setting with a deterministic, *single-output* function f. While f's determinism is just a simplifying assumption, single-output is a necessary restriction for our construction. Indeed, the problem is that a secret-reconstruction mechanism is defined with respect to a single shared value which is reconstructed for all players. However, when evaluating an n-ary function in an MPC setting, the incentive is to make f's ith value only known to P_i. This is impossible for such a generic construction, unless we would introduce new notions for secret reconstruction where n secrets are shared and every party obtains just a single, clearly specified one of these. Yet, introducing truly new notions was not the idea behind our generic construction. Rather we specified the construction generic, in order to instantiate it with existing protocols satisfying existing notions. Hence, in order to construct and analyze evaluation mechanisms for n-ary functionalities, we suggest to

modify an existing reconstruction mechanism adequately to a setting with n shared values, and instantiate Construction 4.1 with it. Of course, in order to analyze its properties, the definitions for games and mechanisms for function evaluation need to be slightly adapted to cover n-ary functionalities.

Bibliography

[Abr+06] Ittai Abraham et al. "Distributed computing meets game theory: robust mechanisms for rational secret sharing and multiparty computation". In: *25th ACM Symposium Annual on Principles of Distributed Computing*. Ed. by Eric Ruppert and Dahlia Malkhi. Association for Computing Machinery, July 2006, pp. 53–62. DOI: 10.1145/1146381.1146393.

[AL11] Gilad Asharov and Yehuda Lindell. "Utility Dependence in Correct and Fair Rational Secret Sharing". In: *Journal of Cryptology* 24.1 (Jan. 2011), pp. 157–202. DOI: 10.1007/s00145-010-9064-z.

[Bei11] Amos Beimel. "Secret-sharing schemes: a survey". In: *International Conference on Coding and Cryptology*. Springer. 2011, pp. 11–46.

[BGW88] Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. "Completeness Theorems for Non-Cryptographic Fault-Tolerant Distributed Computation (Extended Abstract)". In: *20th Annual ACM Symposium on Theory of Computing*. ACM Press, May 1988, pp. 1–10. DOI: 10.1145/62212.62213.

[Bla79] G. R. Blakley. "Safeguarding Cryptographic Keys". In: *Proceedings of AFIPS 1979 National Computer Conference* 48 (1979), pp. 313–317.

[BN07] Liad Blumrosen and Noam Nisan. "Algorithmic game theory". In: *Combinatorial Auction* (2007).

[CDN15] Ronald Cramer, Ivan Damgård, and Jesper Buus Nielsen. *Secure Multiparty Computation and Secret Sharing*. Cambridge University Press, 2015. ISBN: 9781107043053.

[Che15] Mahdi Cheraghchi. *Nearly Optimal Robust Secret Sharing*. Cryptology ePrint Archive, Report 2015/951. http://eprint.iacr.org/2015/951. 2015.

[Cho+85] Benny Chor et al. "Verifiable Secret Sharing and Achieving Simultaneity in the Presence of Faults (Extended Abstract)". In: *26th Annual Symposium on Foundations of Computer Science*. IEEE Computer Society Press, Oct. 1985, pp. 383–395. DOI: 10.1109/SFCS.1985.64.

[DH76] Whitfield Diffie and Martin E. Hellman. "New Directions in Cryptography". In: *IEEE Transactions on Information Theory* 22.6 (1976), pp. 644–654.

[DHR00] Yevgeniy Dodis, Shai Halevi, and Tal Rabin. "A Cryptographic Solution to a Game Theoretic Problem". In: *Advances in Cryptology – CRYPTO 2000*. Ed. by Mihir Bellare. Vol. 1880. Lecture Notes in Computer Science. Springer, Heidelberg, Aug. 2000, pp. 112–130. DOI: 10.1007/3-540-44598-6_7.

[DP13] Sourya Joyee De and Asim K. Pal. "Achieving Correctness in Fair Rational Secret Sharing". In: *CANS 13: 12th International Conference on Cryptology and Network Security*. Ed. by Michel Abdalla, Cristina Nita-Rotaru, and Ricardo Dahab. Vol. 8257. Lecture Notes in Computer Science. Springer, Heidelberg, Nov. 2013, pp. 139–161. DOI: 10.1007/978-3-319-02937-5_8.

[DR07] Yevgeniy Dodis and Tal Rabin. "Cryptography and game theory". English (US). In: *Algorithmic Game Theory*. Cambridge University Press, Jan. 2007, pp. 181–206. ISBN: 9780511800481. DOI: 10.1017/CBO9780511800481.010.

[FKN10] Georg Fuchsbauer, Jonathan Katz, and David Naccache. "Efficient Rational Secret Sharing in Standard Communication Networks". In: *TCC 2010: 7th Theory of Cryptography Conference*. Ed. by Daniele Micciancio. Vol. 5978. Lecture Notes in Computer Science. Springer, Heidelberg, Feb. 2010, pp. 419–436. DOI: 10.1007/978-3-642-11799-2_25.

[Gar+13] Juan A. Garay et al. "Rational Protocol Design: Cryptography against Incentive-Driven Adversaries". In: *54th Annual Symposium on Foundations of Computer Science*. IEEE Computer Society Press, Oct. 2013, pp. 648–657. DOI: 10.1109/FOCS.2013.75.

[GK06] S. Dov Gordon and Jonathan Katz. "Rational Secret Sharing, Revisited". In: *SCN 06: 5th International Conference on Security in Communication Networks*. Ed. by Roberto De Prisco and Moti Yung. Vol. 4116. Lecture Notes in Computer Science. Springer, Heidelberg, Sept. 2006, pp. 229–241. DOI: 10.1007/11832072_16.

[GMW87] Oded Goldreich, Silvio Micali, and Avi Wigderson. "How to Play any Mental Game or A Completeness Theorem for Protocols with Honest Majority". In: *19th Annual ACM Symposium on Theory of Computing*. Ed. by Alfred Aho. ACM Press, May 1987, pp. 218–229. DOI: 10.1145/28395.28420.

[Gol01] Oded Goldreich. *Foundations of Cryptography: Basic Tools*. Vol. 1. Cambridge, UK: Cambridge University Press, 2001, pp. xix + 372. ISBN: 0-521-79172-3 (hardback).

[Gol04] Oded Goldreich. *Foundations of Cryptography: Basic Applications*. Vol. 2. Cambridge, UK: Cambridge University Press, 2004. ISBN: ISBN 0-521-83084-2 (hardback).

[HP10] Joseph Y. Halpern and Rafael Pass. "Game Theory with Costly Computation: Formulation and Application to Protocol Security". In: *ICS 2010: 1st Innovations in Computer Science*. Ed. by Andrew Chi-Chih Yao. Tsinghua University Press, Jan. 2010, pp. 120–142.

[HT04] Joseph Y. Halpern and Vanessa Teague. "Rational secret sharing and multiparty computation: Extended abstract". In: *36th Annual ACM Symposium on Theory of Computing*. Ed. by László Babai. ACM Press, June 2004, pp. 623–632. DOI: 10.1145/1007352.1007447.

[Kat08] Jonathan Katz. "Bridging Game Theory and Cryptography: Recent Results and Future Directions (Invited Talk)". In: *TCC 2008: 5th Theory of Cryptography Conference*. Ed. by Ran Canetti. Vol. 4948. Lecture Notes in Computer Science. Springer, Heidelberg, Mar. 2008, pp. 251–272. DOI: 10.1007/978-3-540-78524-8_15.

[KN08] Gillat Kol and Moni Naor. "Cryptography and Game Theory: Designing Protocols for Exchanging Information". In: *TCC 2008: 5th Theory of Cryptography Conference*. Ed. by Ran Canetti. Vol. 4948. Lecture Notes in Computer Science. Springer, Heidelberg, Mar. 2008, pp. 320–339. DOI: 10.1007/978-3-540-78524-8_18.

[LS10] Anna Lysyanskaya and Aaron Segal. *Rational Secret Sharing with Side Information in Point-to-Point Networks via Time-Delayed Encryption*. Cryptology ePrint Archive, Report 2010/540. http://eprint.iacr.org/2010/540. 2010.

[MSR08] Shaik Maleka, Amjed Shareef, and C Pandu Rangan. "Rational secret sharing with repeated games". In: *International Conference on Information Security Practice and Experience*. Springer. 2008, pp. 334–346.

[Ong+09] Shien Jin Ong et al. "Fairness with an Honest Minority and a Rational Majority". In: *TCC 2009: 6th Theory of Cryptography Conference*. Ed. by Omer Reingold. Vol. 5444. Lecture Notes in Computer Science. Springer, Heidelberg, Mar. 2009, pp. 36–53. DOI: 10.1007/978-3-642-00457-5_3.

[OR94] Martin J Osborne and Ariel Rubinstein. *A course in game theory*. MIT press, 1994.

[RB89] Tal Rabin and Michael Ben-Or. "Verifiable Secret Sharing and Multiparty Protocols with Honest Majority (Extended Abstract)". In: *21st Annual ACM Symposium on Theory of Computing*. ACM Press, May 1989, pp. 73–85. DOI: 10.1145/73007.73014.

[Sha79] Adi Shamir. "How to Share a Secret". In: *Communications of the Association for Computing Machinery* 22.11 (Nov. 1979), pp. 612–613.

[ST05] Yoav Shoham and Moshe Tennenholtz. "Non-cooperative computation: Boolean functions with correctness and exclusivity". In: *Theoretical Computer Science* 343.1-2 (2005), pp. 97–113.

[Yao86] Andrew Chi-Chih Yao. "How to Generate and Exchange Secrets (Extended Abstract)". In: *27th Annual Symposium on Foundations of Computer Science*. IEEE Computer Society Press, Oct. 1986, pp. 162–167. DOI: 10.1109/SFCS.1986.25.